FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Fares Calculator over a Public Transportation Network

Joel Ramos

Mestrado Integrado em Engenharia Informática e de Computação

Supervisor: José Luís Borges

Proponent: Optimização e Planeamento de Transportes, S.A. (OPT)

July 25, 2014

Fares Calculator over a Public Transportation Network

Joel Ramos

Mestrado Integrado em Engenharia Informática e de Computação

Approved by:

President: Prof. António Miguel Pontes Pimenta Monteiro

Referee: Prof. Maria Benedita Campos Neves Malheiro

Referee: Prof. José Luís Borges

July 25, 2014

Abstract

Nowadays, in existing Public Transportation Networks, public transports operators usually have a system that allows route calculus on that operators transports. Price calculations are sometimes included. However, for the user, there is no simple method to calculate rates inside geographical locations where multiple operators co-exist. Since recent studies show that good public transports are on top of younger generations when searching for a city to move in, a tool that is able to make such calculus must be of value.

When traveling on paid public transports a user needs to buy a ticket either on the vehicle itself or a ticket bought a priori. When traveling in a trip where only one fare is applied the calculus of the price is a simple operation, or it should, that requires only consulting the information available by the providers. However on trips that have multiple fares being applied there isn't a single place that combines all of the information that allows a user to make calculations to find best combination of tickets he should buy for his trip to be as cheap as possible. Even if such information exists centralized on a single place, the calculus of the several options available and of the cheapest combination of tickets may be hard for a user to make.

The tool described on this document to solve this problem aims to be integrated into OPT's system of retrieving public transportation trips in order to provide users with information, regarding a given trip, of its fares and necessary tickets optimizing the total cost of the trip.

This document describes the research for public transportation fares and the implementation of system that is able to store fares from any public transportation network to use them for the calculus. To test the feasibility and analyze its performance two case studies where used: one with the network from Porto and another randomly generated using a model that describes a real-word public transportation network.

As it will be shown on this report the algorithm scales well and so it can be applied to any network regardless of its size.

Acknowledgments

Firstly, I would like to thank to my supervisor, professor José Luís Borges for his help, specially on the advices given to write this report.

I would like to thank Eng. Luís Filipe Ferreira, my supervisor at OPT (Optimização e Transportes S.A.), to Dr. João Marques and Eng. André Dias, my working colleagues that have been a great help through the development of my, and to my friend João Anes that shared with me the pleasure of being at OPT working at thesis.

The most sincere thanks to all my friends, specially from my course, with whom I've shared many hours while writing this report.

Finally I would like to thank my parents, Adelaide and Américo, the most responsible people for me to be writing this thesis, and to my brother Celso, always a person worth debating about any subject of any kind even the ones from my thesis.

Joel Paraíso Ramos

"One man's "magic" is another man's engineering.
"Supernatural" is a null word."

Robert A. Heinlein

Contents

List of Figures

Abbreviations

PTN	Public Transport Network
OPT	Optimização e Planeamento de Transportes, S. A.
CP	Combois de Portugal, S. A.
APSP	All-pairs shortest path problem
IMS	Information for Mobility Support
ANTRAL	Associação Nacional dos Transportadores Rodoviários em Automóveis Ligeiros
FPT	Federação Portuguesa do Táxi
STCP	Sociedade de Transportes Colectivos do Porto, S. A.
ES	Espírito Santo
ETG	Empresa de Transportes Gondomarense
WCF	Windows Communication Foundation
ORM	Object Relation Mapper
PFF	Provider Forecast Feeder
CRUD	Create, Remove, Update and Delete
UML	Unified Modeling Language
XML	eXtensible Markup Language
XSD	XML Schema Definition
TfL	Transport for London

Chapter 1

Introduction

The research for optimal routes on networks is a popular problem in the field of graph theory. The application of that problem to Public Transports Networks (PTN) lead to the appearance of journey planners. Journey planners are applications, usually online or for mobile decies, that help users scheduling their trips by allowing users to input a start and ending point of a trip and returning a full plan for the journey between those two points. In spite of many of the services providers (such as buses, trains, subways) supplying such applications for their own networks, it is unusual to find applications to plan trips inside a given area regardless of the providers the journey may use. Even more unusual is to find journey planners for areas that inform the user of the cost his trip will have.

The reason for the lack of pricing information on journey planners is usually connected to the fact that each provider have its own fares with its specific rules for pricing trips and so it is hard to combine all these fares together in a system. There is not a specific study to find a model to encompass and generalize the existing fares and a calculator able to return prices for trips that use one or more of those fares. There are some studies on this thematic but they focus only on fixed-charged transportation network [1].

The previously mentioned studies not only ignore the fact that most Public Transportation Networks are now using zone fares, which do not guarantee that fares are fixed, but also the fact that in the biggest cities Public Transport Integration is becoming a standard. In spite of that standard sometimes it's possible to find some examples of multiple operators providing services in the same area [2, 3] that do not use the same fare system.

1.1 Context

Optimização e Planeamento de Transportes S.A. (OPT) offers a service, called MOVE-ME, available both on mobile (for Android and iOS) and on a Web application focused on finding routes inside the PTN of Porto[1] and Lisbon[2]. Besides finding the ideal routes for a user to take in order to reach his destination in less time, it also providers information on the nearby stops using the

[1] http://www.move-me.mobi/
[2] http://lisboa.move-me.mobi/

mobile application geolocation features and the estimated time services are expected to arrive at those stops. This system does not provide pricing information for the trips it calculates.

Recent studies point that a good transport system in a city is on the top of the priority list when the younger generations look for a city to move in. Given the broad usage of smart phones by those people, it is expectable they look for applications like MOVE-ME to route their trips through cities and, so, a feature that returns the price of the trip is expected.

It is important, firstly, to define what a trip is. A trip is a set of one or more segments connecting a start and an ending point between where someone intends to travel. These segments may all be provided by the same transportation company, the providers, or may have distinct providers, means of transport and fares associated.

If finding the price of a trip that uses a single fare may appear as a trivial operation of consulting tables and the information regarding that fare, when a trip uses more than one fare or multiple fares for the same path are available the complexity of the operation increases. If single segment trip between two points (with no transfers) has more than two fares available to be used, users will calculate the two prices and choose the cheapest.

Some fares are known to have tickets that are not payed by the service used but are prepaid and have a limit of time to be used. For trips with transfers, or in another words trips with more than one segment on its path, a user may be lead to question if his ticket will still be valid on the segments other than the first. Another possibility the user may question himself is if the change to a second mean of transport, either it's from the same type and from the same provider or not, implies an extra price to be paid. If that same user is traveling on services where multiple fares apply what is the best option: to re-use the same ticket or to buy a new one?

This questions are dependent on the type of fare implemented by each provider and on some networks they may be frequent and in another this cases may never occur, which raises a main research question:

> What fares exist for Public Transport Networks and how can a model describe all of them? Can a generalized calculator fare tool that fits any network be developed?

A calculator tool to be integrated into OPT's system was developed and it will be described on this document along with the research necessary in order to develop it. This tool will allow to add the feature of consulting the price on MOVE-ME not only complementing the available information but also creating a new use case: the search of a trip in order to check its price.

The project described on this document was proposed by OPT and developed under a master thesis dissertation of the Master in Informatics and Computing Engineering from the Faculty of Engineering of the University of Porto. The development took place at OPT's headquarters between February and June of 2014 and supervised by Eng. Luís Filipe Ferreira, from OPT, and the professor José Luís Borges.

1.2 Motivation

The increase of population, especially in major cities, the massive transit generated in the big cities, fuel prices and a more conscious and worried society about the environment is driving to the increase usage of public transports. At the same time the idea of liberalization of the transports service market to allow fairer competition - established by the European Union with a deadline of 2019 [4] - will, in theory, lower the prices at the market [5].

In spite of this growth in usage and the expectation of a more and more open market to drive prices down, there is no tool widely available to calculate prices of public transports and to allow a user to freely choose the cheapest. Thus this tool is expected to be able to have a real impact on peoples lives.

The work developed for this dissertation will allow to create such tool and, hopefully, try to accelerate the implementation of such system in Portugal contributing for a more technological evolved country.

Another motivation with the possible outcomes of this research is that the results of this kind a system may help public transports operators to further studies on the constitution of their own networks and possibly to help them to erase possible inequalities on trips costs among users in a network.

1.3 Dissertation Structure

Besides this introduction this report contains six other chapters.

Chapter 2, the literature review, describes the items researched that were required to develop the work presented on this report. An explanation of the proposed problem and an overview of the Web stack where the proposed system is intended to be integrated is described at chapter 3.

The proposed solution, or hypothesis, can be found at chapter 4 along with a description of the system's implementation and its performance analysis.

Chapter 5 describes the case studies used to validate the work done.

A brief discussion and comments about the work produced can be found on chapter 6 and the chapter 7 concludes this report.

Appendixes of this document contain XSD definition of the XML files that load the fare information into the system that was developed (appendix A), the results of the Case Studies that will be presented during this report (appendixes B and C) and the fares from the Porto Metropolitan Area that exist in the OPT's system and used for the Case Study of the Porto Network.

Chapter 2

Literature Review

The following chapter describes the research made to be able to develop the proposed work. Research on the characteristics of public transportation networks and fare system is included as well as an explanation of the Andante fare. Also included on this section is the research made on routing algorithms necessary to understand the way the algorithms that were already implemented on the Information for Mobility Support (IMS) system works.

2.1 Public Transports

Public Transports designation applies to shared transports of general usage by the population. In metropolitan areas one can find buses, trolleybuses, trains, subways and ferries as means of transport and find airlines, coaches, intercity and high-speed rails to transportation between cities or metropolitan areas. These kind of transports have in common:

- A known service with known temporal and spacial cost;
- Points in the route that can be defined as boarding or landing spots;
- Regular services having a known timetable.

So, theoretically speaking, its possible to define a model that can accept and integrate all Public Transport services existent in a given area to allow fare calculations and, as a result, to be able to minimize time and money for the user when planning a trip.

2.1.1 Transports Network

A Transports Network can be defined as a network containing services and infrastructures that allows for vehicular movement or flow allowing for populations to travel inside a predetermined area. Such network may have multiple types of transport's vehicles like the ones that travel using

road network (like trams and buses), that move through railway network (like subway, train, high-speed rail...), that move through water (ferries, ocean liners and other kind of watercrafts) and aircrafts.

Taking aside taxis and rental services, that use the existent road network and charge for their services by the proportion of time spent and/or distance traveled and have pick-up and drop-off locations that vary, all the other means of transportation for general usage of the population use well defined boarding and landing places and infrastructures. This means that the duration of a trip, the distance traveled and the route taken by public transportation services can always be predicted unless events like traffic congestion, road blockages or bad weather occur. Taxis and rental services are commonly referred as Private Transports as opposed to the remaining examples presented which are Public Transports - a shared passenger transport service available for general use of the community unlike the Private Transports which are not shared by strangers without a private arrangement.

The fixed infrastructure and predictability of Public Transports allows the calculus of the price of a trip a priori. To make such calculus some providers like Metro do Porto offer a web application where the user input's the start and the end of the trip and a route, and its respective price, is calculated.

To calculate the costs of Private Transports there are also some options available for users, such as the usage of online mapping services like Google Maps or OpenStreetMap, that allow to calculate distances and travel times in road networks. The price charged by this kind of services is commonly regulated by the countries authorities. In Portugal those prices are regulated by a convention made between the authorities (Direção-Geral das Atividades Económicas) and by two taxi drivers organizations (Associação Nacional dos Transportadores Rodoviários em Automóveis Ligeiros (ANTRAL) and Federação Portuguesa do Táxi (F.P.T.)) [6].

2.1.1.1 Public Transport Networks properties

The evolution of Public Transport Networks over time is closely related to the growth of the city itself and therefore by social, historical and geographical factors from each city. In spite of this high amount of variables that influence the network itself, Public Transport Networks on different cities share common statistical properties due to their functional purposes. [7].

As observed by some authors, many of this networks representations tend to follow a power-law distribution and, so, they can be defined as scale-free networks [8, 7, 9]. A scale-free network is a network that has a power-law degree distribution, i.e. probability distribution of the number of edges adjacent to each node follows a power-law, which can be expressed mathematically by the equation 2.1:

$$P(k) \sim k^{-\gamma} \tag{2.1}$$

One of the reasons pointed as why Public Transport Networks are of such type is because of the objectives to optimize their operations since scale-free networks have been shown to arise when

there is an optimization to minimize the effort for communication and for maintaining connections [10, 7].

However, and although everyone has an intuitive idea about the concept and what a Public Transport Network is, there are numerous ways to define its topology and therefore numerous representations the graph representation the network may have[7].

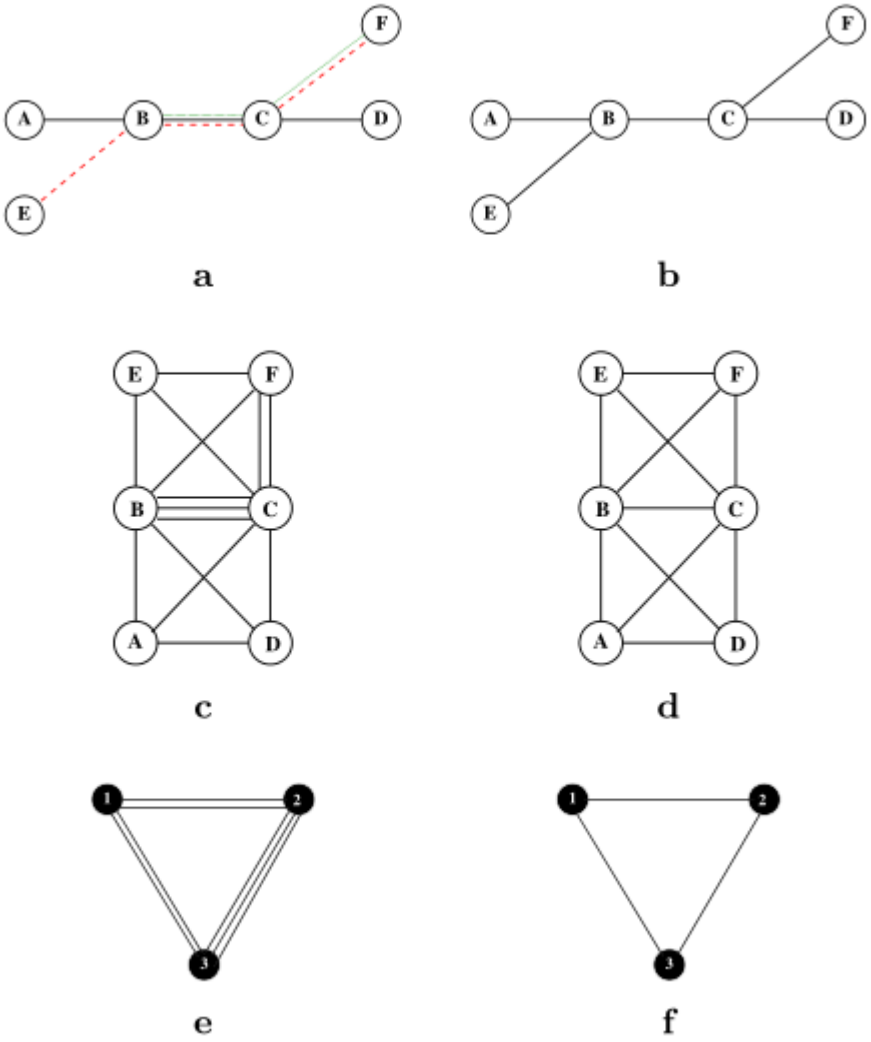


Figure 2.1: Possible graph representations of Public Transport Networks (Source: [7])

Figure 2.1 illustrates a small piece of a public transportation map. Stations A-F are serviced by the train no. 1 (in a solid line), train no. 2 (dashed line) and train no. 3 (dotted line) and the map is represented in 3 different ways.

Figure 2.1a represents each station by a node and a link between each node indicates a service between those two nodes. Figure 2.1b is the same representation but each edge represents at least one route between services ($\mathbb{L}$ representation). On the representation of figure 2.1c all nodes that

belong to the same route are connected and in figure 2.1e each route is represented by a node and each link corresponds to a common stop shared by the route nodes it connects. Figures 2.1d and 2.1f are the same representations but the edges of the graph represent one or more routes ($\mathbb{P}$ and $\mathbb{C}$ representation respectively).

C. von Ferber found that according to the type of representation of the network graph some real-word public transportation networks may cease to be scale-free networks and its node degree follows an exponential decay distribution[7][p. 18] which can be represented by the following mathematical expression:

$$f(x;\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases} \tag{2.2}$$

His studies on multiple cities showed that most of the networks when drawn in the $\mathbb{L}$ representation follow the power-law and some networks when drawn in the $\mathbb{P}$ representation follow the exponential decay distribution but he could not find any explanation for this.

2.1.2 Fare Systems

Each Public Transport Operator has a fare system defined for their services. This can be a system where ticket rates are fixed for a journey or one where the same is proportional to the time spent on the transport or to the distance traveled. An example of one of this fares is the one used by Rodoviária do Tejo, a public provider operating in the area of Leiria. The prices practiced on each trip are set according to the distance between the starting and the ending point[1].

The implemented system can also be a zoning system where the cost of a trip is calculated by the amount of zones that the user crosses. However, both type follow the same basic objectives in common [11, 12]:

- Simple and clear tariff systemx
- Standardized ticket assortment
- Uniform tariff regulations
- Constant price differences between zones (for zone tariffs)
- Limitation of price hikes at zone boundaries (for zone tariffs)
- Declining tariffs for passes
- Linear tariffs for single and multi-trip tickets

The same authors say that there are three main structure models of zoning that can be followed to create and implement tariff systems in a Transportation Network on cities:

- Ring Structure
- Area Structure
- Honeycomb Structure

[1] http://www.rodotejo.pt/informacoes/tarifarios

The ring structure is constructed by the overlap of rings across a central area. This model of zoning is characteristic of areas where the core of the metropolitan area is at a given center. The rings area created by progressive increases of the radius as they depart away from the center. Figure 2.2 shows an example of a ring fare zone, the Lisbon zoning for monthly travel cards.

Figure 2.2: Tariff System in Public Transport in Lisbon

The area zones structure is a very similar model to the ring structure's. This model combines characteristics of the ring model and the honeycomb structure, mentioned ahead on this document. The zones are, as the previous model, constructed having a central zone on the core of the area and then rings are overlapped across that area. The difference between these two mentioned models is that the rings around the central area of area zone structure are subdivided into sectors that try to match the main transport corridors of the region. The Amsterdam tariff system, illustrated in figure 2.3, is an example of such zoning model.

Finally the honeycomb structure is a zoning fare that is based on a multi-directional zoning where the region is covered with a grid where each space is a different zone. This model characteristic that distinguishes it from the other mentioned models is that it allows a better match between the distance traveled and the charged tariffs. The zoning system from the Metropolitan Area of Porto, the Andante, is an example of this model as illustrated in figure 2.5.

Each of this zoning models have several advantages and disadvantages and their implementation in the cities most of the times depends on the geography of the terrain itself. For instance Lisbon has geographical characteristics that allows to define a center and to create a set of surrounding rings around this center in a close to circular form.

The traditional way that tariff rates are calculated for the user is through the count of the number of zone blocks crossed. The user, to travel between two points in the network, it is required to have

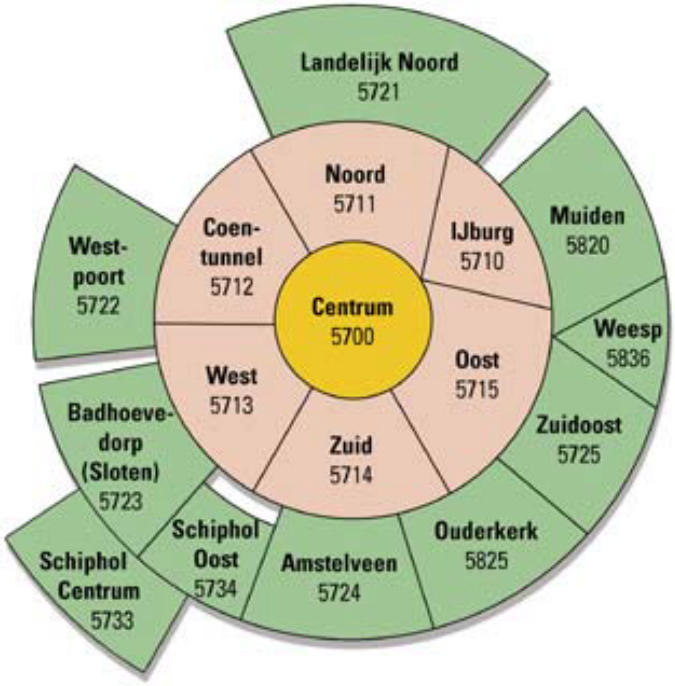


Figure 2.3: Tariff System in Public Transport on Amsterdam

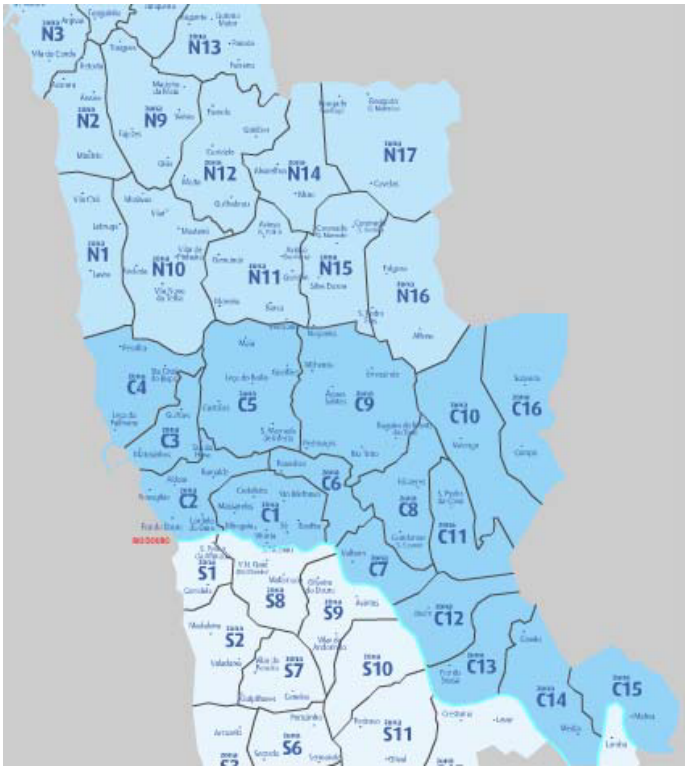


Figure 2.4: Andante Tariff System in the Metropolitan Area of Porto

a valid ticket for all zones crossed.

It's practice, specially on honeycomb structured fares, that the first border crossover is not counted in order to create justice for the users who make journeys that start close to the border of the zone and need to cross them over. In other words the base price of a trip allows for travel in the first two zones.

There are also cases, like the services managed by TfL (Transport for London), where zoning models area applied but the prices are set by a table that defines prices from zone-to-zone and which include peak and off-peak fares as well as daily maximum amount to be paid by the user, discounts and other restrictions [13, 14].

2.1.2.1 Porto Metropolitan Area Fare System: Andante

The fare Andante is an intermodal fare system active in the Metropolitan Area of Porto that combines the fares of several transports and operators from this area. With a ticket from this fare a user can ride the subway system of the system, Metro do Porto, use it for partial segments of some train lines, from CP - Comboios de Portugal, and travel on the following bus providers: STCP, Resende, Espírito Santo (ES), Maia Transportes, Valpi, Empresa de Transportes Gondomarense (ETG), MGC Transportes, ANC and Auto-Viação Pacense.

Besides STCP, the biggest provider on this area that operates mostly on the city of Porto, the other bus companies services are only partially covered by this fare.

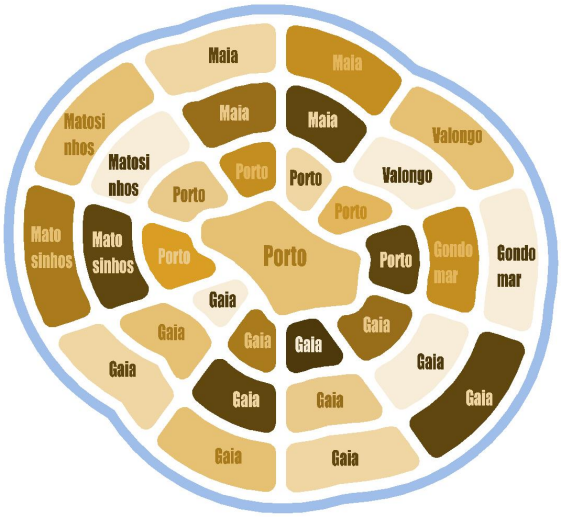


Figure 2.5: Andante Tariff System in the Metropolitan Area of Porto (Source: [15])

Due to the fact of encompassing the Metropolitan Area of Porto, it is not possible to define a center fr the Andante fare due to the number of metropolis on this area. The implemented model on this tariff is the Honeycomb Structure.

In order to calculate the price to charge the user in an occasional ticket on Andante, there is a combination of the traditional calculations mentioned on the previous section and some characteristics of the ring zoning structure. Being so, the price charged for a title of traveling

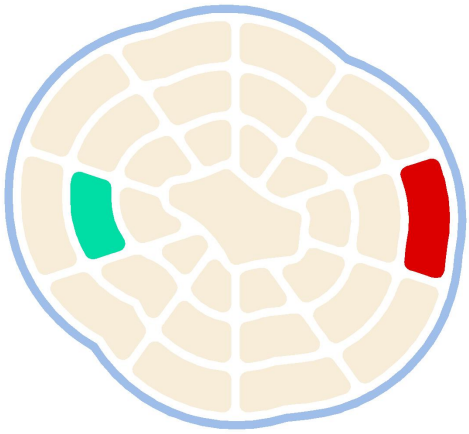


Figure 2.6: Zone of origin and Zone of destination of the sample trip (Source: [15])

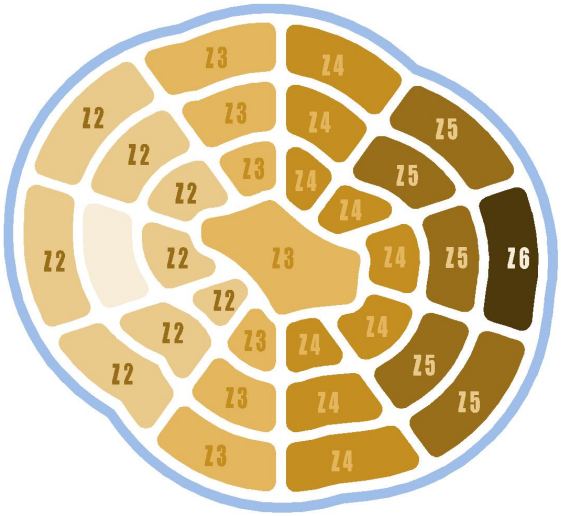


Figure 2.7: Type of ticket necessary to the sample trip (Source: [15])

between two zones is calculated using the starting point of the journey as the origin and incrementing that price whenever the user crosses the board for the next set of rings that surround that zone. This calculations applies recursively between the start and the finishing zone of the trip.

Figures 2.5, 2.6 and 2.7 represent a schematic example of a trip between Matosinhos and Gondomar and the type of tickets applied. Figure 2.5 represents the network of Andante drawn in a schema and figure 2.6 shows the starting point, the zone colored at the left of the picture, and the ending point, the zone colored at the right, of the example trip. As it can be seen in figure 2.7, through the color gradient of the zones around the starting point, the increment in the tariff is done by counting the set of rings of zones surrounding the zone of the starting point. It can be said that the count of zones on this tariff system is like placing a ring zone structure centered on the beginning of the trip. Although the calculation system is more advantageous for the user, given that it allows for him to travel in a broader area, it can lead to caricatured situations like the Metro trip between Póvoa do Varzim and Hospital de S. João that have different prices depending on the way of the service, like its shown in figure 2.8.

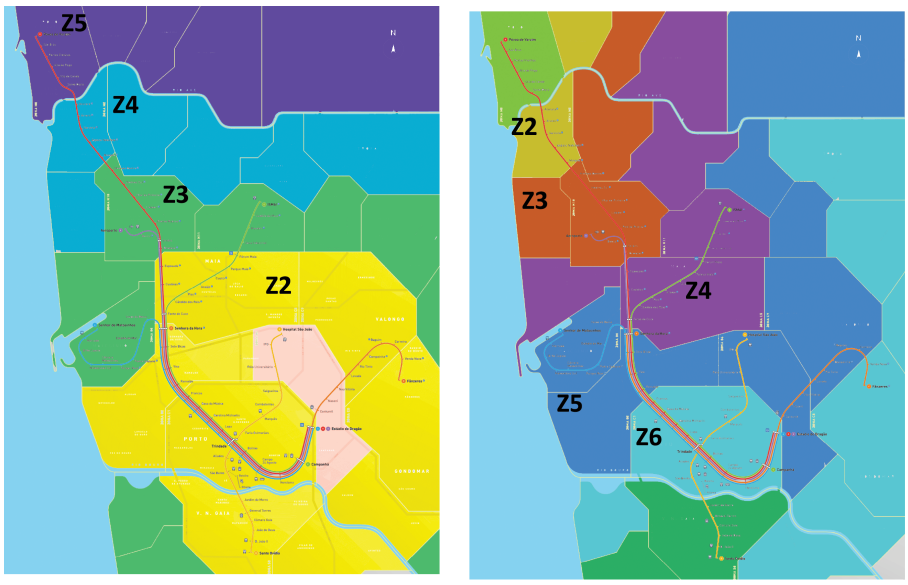


Figure 2.8: Example of the necessary ticket type for the trip Póvoa do Varzim -> Hospital de S. João

2.2 Routing Algorithms

Routing is the process of selecting the optimum path in a network. As Léon Planken stated, path finding can be applied to all sorts of networks, such as roads, water, electricity, communications and computer networks and so, the number of algorithms developed along the years is immense.

Although the focus of the dissertation is not directly related to routing of trips itself, a good knowledge on this thematic was helpfull to understand the way the algorithms implemented at OPT's systems works.

2.2.1 Shortest Paths Algorithms and their performance analysis

When applying a shortest path algorithm to a generic public transports network one must be expecting a large network where the algorithm performance is crucial and, so, simple implementations of shortest paths algorithms like Floyd-Warshall algorithm, Johnson's algorithm or the classical approaches of Dijkstra or A* search algorithm's (for single-source shortest path problem) won't be the best option.

Ravindra K. Ahuja et. al [16] and Stefano Pallottino et. al [17] work on this area of research are the most accepted as the best on levels of performance both theoretical and computational[18, 19]. Both of this algorithms are based in a implementing a *heap* structure to perform a better sort algorithm. According to Ravindra K. Ahuja et. al [16] "The key to efficient implementation of Dijkstra's algorithm is the use of a data structure called a *heap (or priority qluele)*. A heap consists of a set of items, each with an associated real-valued key, on which the following operations are possible:

insert(h, x) Insert new item x, with predefined key, into heap h

delete min(h) Find an item of minimum key in heap h, delete it from h, and return it as the result of the operation.

decrease(h, x, value) Replace by *value* the key of item x in heap h; *value* must be smaller than the old key of x

In a heap-based implementation of Dijkstra's algorithm, a heap h contains all the labeled vertices; the tentative cost of a labeled vertex is its key." [16, 18, p. 214].

Although there is this similarity, the two algorithm's "differ according to the rules used to select labeled nodes for scanning and in the data structures used to manage the set of labeled nodes" [19, p. 67]. Although the re-distributive heap algorithm, Ahuja's, has the best theoretical efficiency in larger networks [18, 16, p. 22] ($O(m+nlog(C))$ complexity opposing to a $O(n^2m)$ complexity for Pallottino's algorithm [19, p. 67]), the Pallottino's appears to have better performance in computational result of smaller networks and an identical result to Ahuja's algorithm [18, p. 30] in larger networks. F. Benjamin Zhan et. al [19] also refer that an algorithm based on the Pape-Levit implementation is as fast as the Pallottino's although he admits that Pallotino's has a slight edge [19, p. 72] and, also, Andrew V. Goldberg encountered some variations in Pape-Levit algorithm's performance since it has poor time bounds [20, p. 6].

Pallottino's algorithms implementations can than be accepted as the best performing for solving the one-to-all shortest path problem [19, 18, p. 72, p. 31], the public transport network problem, specially since it was tested and accepted as such on both a computation study using a network generated from real roads from 10 states across the Midwest and Southeast of the United States [19, p. 72] and also proved to be efficient in smaller networks [18].

Arc length appears to have a crucial in the performance of the algorithm and so it is worth studying ways to minimize it [19, p. 71].

The all-pairs shortest (APSP) path problem is also extensively studied but its complexity has remained open to this day [21]. Chain's final remarks on a revised version of the previous cited paper ends with a (still empirical) question: "can the general APSP problem be solved in $O(n^{3-\delta})$ time for some positive constant δ"?.

The best known APSP algorithm, as presented in ESA'06 Proceedings of the 14th conference on Annual European Symposium , for the all-pairs shortest-path carries a complexity of $O(n^3(log\frac{log(n)}{log^2(n)})^{\frac{5}{4}})$ [22]. Wei Peng et. al [23], at 2012, claims that his proposed algorithm reaches experimental results that improves the theoretical complexity of $O(\frac{n^3}{log(n)})$, proposed by Chan [21] and that seems to be proved with experiences by Yijie Han[22], to a complexity of only about $O(n^{2.4})$.

2.3 Decision Making

As defined on the Oxford Dictionary decision-making is "the action or process of making important decisions" and a decision is "a conclusion or resolution reached after consideration". One of the most powerful methods of generating decisions in computer science are the decision trees. Since there is the necessity two calculate multiple prices on some trips where multiple fares co-exist, a decision tree may be helpful for this operation.

2.3.1 Decision Tree Model

A decision tree model is a model of computation in which an algorithm is considered to be implemented as a decision tree. It is a decision support tool that uses a graph built as a flowchart to represent different strategies in order to reach a certain goal. It consists on 3 types of nodes:

Decision Nodes commonly represented by a square, decision nodes are the nodes where a decision has to be made in order to reach a goal;

Chance Nodes this nodes are usually represented by a circle and they are associated with decisions that have a probability associated to reach each goal represented as the children nodes of the chance node;

End Nodes end nodes are usually represented as a triangle and they are the the leafs of the tree; this nodes can be classified as the possible goals of the decision tree.

Since fares in Public Transport Networks are always known, when modeling the problem of finding the right fares to be used on given trip as a decision tree there are no chance nodes represented. Another characteristic is that since the ideal fares for a given user is a sequence of fares, active between the begin and the ending of the trip, the tree is divergent and has always a root node, which is the starting node of the trip.

To solve this problem four basic data structures were found on the researched literature that had usages similar to one intended to solve the problem of this dissertation: segments trees, interval trees, range trees and b-trees. The paragraphs that follows describe such trees.

A segment tree structure is a data structure to represent intervals whose endpoints are fixed or known a priori. Each node contains, besides its value and key, a range of integers $[B, E]$ representing a range of indexes from B to E, a key for splitting this range into two subranges each associated with each child of the node and two pointers to the subtrees at the left and at the of the node. Segment trees are optimized to search for points contained in the intervals stored on the structure [24].

An interval tree is a data structure similar to the segment tree. Its nodes, besides containing its value and key, contain two pointers for the subtrees at the left and right [24]. This trees are similar to segment trees but are optimized to search for intervals that overlap, within the intervals stored on the structure.

Range trees are data structures optimized to store a list of points. This trees solve the "range search problem" which is to find all points on the set represented on the tree that satisfy any range query [24]. Range trees can be of dimension k which means that the problem can be solved not only in 1D dimension (which is simply ordering and array and search the array for the given range) for for k ranges. Figure 2.9 is an example of a range tree with a dimension $k = 2$.

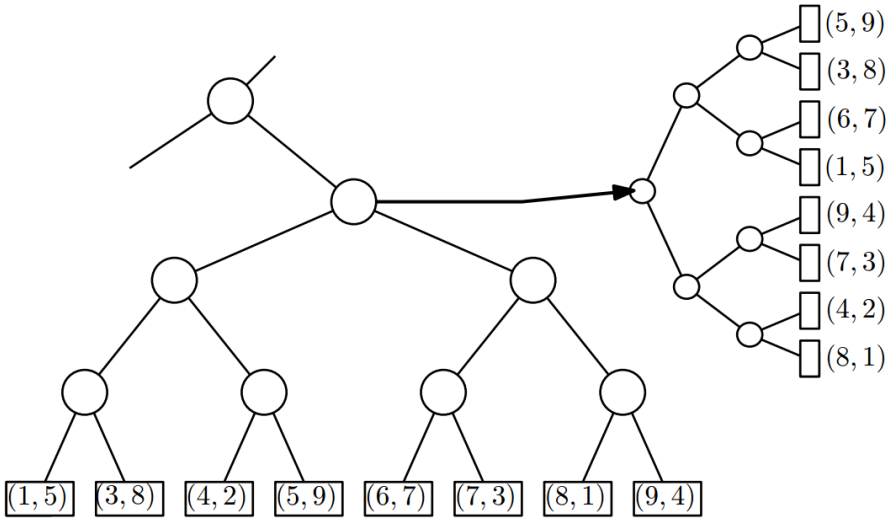


Figure 2.9: Sample of a range tree with dimension $k = 2$ (Source: https://www.cise.ufl.edu/class/cot5520fa13/CG_RangeTrees.pdf)

A b-tree is a generalization of the well known binary-search tree (BST) [25] where a node may have more than two children. Each node has a predefined range of child nodes instead of the maximum of the children nodes imposed by the BST. B-Tree keeps the data stored and allows searches in logarithmic time. This trees are usually used for large data storage and for file organization usually for sorting and searching algorithms [25].

Range trees where found to be the closest solution to help the decision making process required and so its construction algorithm were followed.

Chapter 3

Problem Statement

Optimização e Planeamento de Transportes S.A. (OPT) is a company aimed at developing and deploying intelligent and automatic solutions for decision support in transport planning. IMS (Information for Mobility Support) is a solution stack to store, manipulate and retrieve information regarding public transports in Porto and Lisbon. Currently the information available is given by OPT's strategical partners, the public transports operators, from the previously mentioned areas.

The existent problem was the lack of a tool able to calculate and retrieve information of fares that can be applied to a given trip thus the necessity of a Fare Calculator emerged. The primary purpose of the Fare Calculator is to be integrated into the IMS. This integration allows any service inside the web stack to make calls to the service.

The following section describes the proposed problem and the web stack where the tool is going to be integrated.

3.1 Calculating fares in Public Transport Networks

Public transportation plays an important role across the globe. It's importance on reducing gas emissions to improve the environment and its social importance on ensuring that all members of the society are able to travel even without a drivers license like the young, the old, the poor, people with medial conditions...

There is no easy or convenient way for people moving out of their living or working area for finding the prices and fares associated with transports in that new area. The traditional ways the users from public transports use to find those prices are by searching for printed information on the stops of the public transport (like on the bus stops, the subway entrances, airport information desks), which are often out-of-date, or by going to either the main interface stations or to the tourist offices in the area and request information about the prices and timetables on the network.

In an increasingly technological world this is not the best solution.

Some public transportation providers have web applications, or even mobile applications, that generate routes on their network, containing information about the transports timetables and, sometimes, their fares. From the analysis made on the websites of the biggest providers of public

transports in Portugal, STCP and Carris for the buses and Metro do Porto and Metro de Lisboa for subways respectively in Porto and Lisbon, it was possible to verify that all of those websites contain information regarding the fares, like the information found on their main offices, and a route builder to build routes between a start and ending point inside their networks and using their own services. However, from those four websites, only Metro do Porto contains information of the price of a trip build using their route builder.

For people travelling in those cities the lack of the price when building a route for their trips on those websites can be a major obstacle. Since each operator only offers information regarding their own networks information that can be relevant for the user's route may be omitted. OPT's response to overcome this issue was to create the MOVE-ME system, an intermodal journey planner. MOVE-ME is an application that gives users information about the network services of a given area - currently from Porto and Lisbon - and builds routes inside those areas in order to return the best route for a user to take, based on the chosen optimization, regardless of the provider it chooses. Sometimes the route to a destination can comprehend a mixture of several trips from different providers. Although this goal has been achieved, the issue of knowing the price of a trip wasn't solved yet.

Calculating the price of a trip, or a set of trips, from a provider with one active fare is usually a simple operation to be done by a computer since it is a matter of applying the set of rules for the fare the provider uses. These rules can be, for instance, a pricing according to the count of zones where the trip takes place or the query of a table containing fixed prices for each trip. When multiple providers are found in on built route the usual approaches on this problem is to reduce the fares to one that is common to all the used providers. Although this is the ideal situation, both for the users and for the providers, the reality in which we live is not confined to this theoretical scenario.

When a provider has more than one active fare, or when multiple providers co-exist but there is no common fare on a built route, the calculus of the best price for a trip is no longer simply applying the defined rules for one fare. In this case there will be steps of the trip where the user will face a decision of choosing the right fare. This decision may vary due to several motives: the user may already have a valid ticket that he has used on a trip before; a more expensive fare on a given point of the trip might be better in the final price of the entire trip; it might be better to choose a fare that, further on the trip, still maintains in spite of one that is valid only for that portion of the trip...

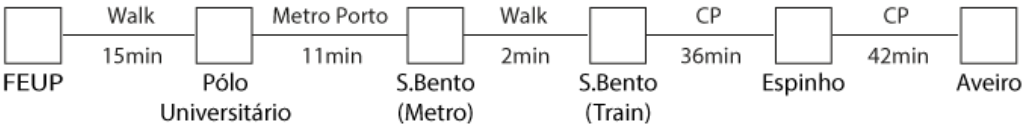


Figure 3.1: Schematically representation of a trip between the Faculty of Engineering and Aveiro, with transfers on S. Bento and Espinho

Let's take a look at the trip schematically represented on figure 3.2 for a sample of a trip containing multiple fares on its path. This trip has its starts at the Faculty of Engineering (FEUP) and ends at Aveiro. The fastest route for a user to achieve the destination is by walking up to the

subway station “Pólo Universitário”, ride the subway until the “S. Bento” station, switch at to the same named station to ride the train up to “Espinho” and then proceed, on the same train line, to “Aveiro”. For the sake of this example let’s assume the user leaves the train at the “Espinho” railway station and waits for the next train.

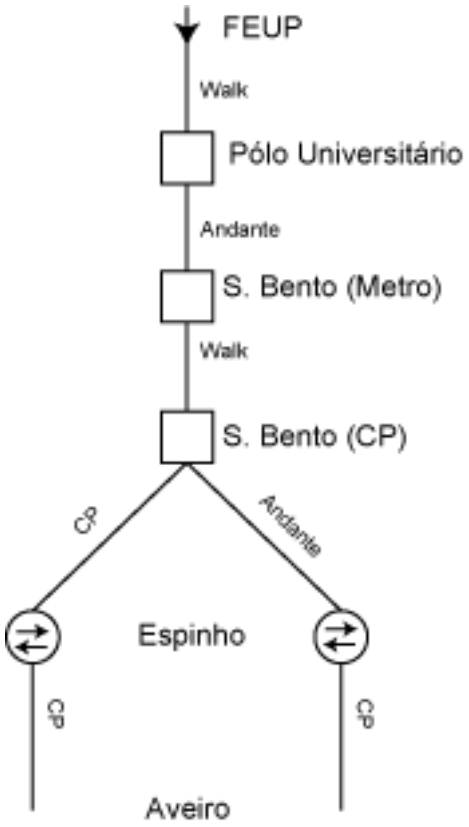


Figure 3.2: Tree representation of the fares for a trip between the Faculty of Engineering and Aveiro, with transfers on S. Bento and Espinho

Figure 3.2 lists, in the form of a tree, the fares the a user encounters when doing the trip illustrated on figure 3.2. Before proceeding to the calculus of the best price and its respective tickets for this trip there are some considerations worth mentioning about the fares active on the route (“Andante” and “CP monomodal”):

- Andante is a multi-modal fare active on the metropolitan area of Porto. A deeper description of the characteristics of this fare can be found at section 2.1.2.1 and its pricing may be found on appendix D;
- There are several levels of tickets in the Andante fare (Z2, Z3, Z4, Z5...) each of them with a different number of zones the user is allowed to travel on and a maximum amount of time permitted. These values increase as the level increases. A unlimited number of transfers can be made using the same travel card as long as the time between the first and the last validation does not exceed the the limit;

- The fare implemented by CP (Comboios de Portugal) on their trips is based on a table that assigns a zone number for a trip between two points. To each zone there is a corresponding price which can be consulted on appendix D;
- Although CP has a rechargeable travel card for occasional trips this card is valid only for only a single path and has no temporal restrictions, being only valid for one validation and one trip. In practical terms

The next paragraphs describe how the problem described before should be solved, step-by-step, in order to return the prices and tickets for the example trip.

The first segment of the example trip is a walking segment between FEUP and the station "Pólo Universitário". This segments may appear sometimes on the trips. Whenever a walking segment is the first segment or the last segment of a trip it should be discarded since it has no influence on the produced outcome. If the segment is encountered further ahead on the trip, in its middle, they are important to be processed since the time of walking between two stations is a factor that makes a difference when validating a ticket for the second time. Thus the first segment of this example trip is going to be ignore.

Between "Polo Universitário" and "S. Bento (Metro)" only the Andante tariff is active and so the user mandatory has to use tickets from this fare. The fare will be stored on the list the list of fares and, along with it, the time of validation and the segment where it occurs. The next segment is a walking segment and so the only influence it has on the algorithm is the increase of time already spent on the previously stored fare.

When reaching "S. Bento (CP)" there are two possible decisions: either the user maintains his ticket from Andante and proceeds or he buys a ticket from the CP's fare. Since that at this point there is no information that can support which is the best decision there is a necessity of calculating both of the possible outcomes.

The alternative on the left, after the"S. Bento (Metro)" decision node of the tree representation of figure 3.2, is a trivial case to be calculated since that, as stated before, the CP's fare is only valid for a single path. The fares for the next two segments of the trip ("S. Bento (Metro)" $\rightarrow$ "Espinho" and "Espinho" $\rightarrow$ "Aveiro") will be added to the list of tickets.

However the alternative on the right requires some processing since a fare that has already been used before on the trip appears again. Some questions rise concerning the tickets of that given fare:

- Can the ticket be used for more than one service?
- If so, is there any extra price for using a second service or no change on the price is made?
- Is the ticket used before still valid or did it exceed the maximum time of usage?
- If it exceeded the maximum time, is it cheaper to buy a new ticket or should the first ticket bought have been of a higher level?

After these steps to store and identify the sequence of fares, a calculus for each sequence should be done to solve the previous questions in order to obtain the cheapest prices for each sequence and to be able to tell the user which of them is the optimal solution.

3.2 IMS - Information for Mobility Support

The IMS system is web stack composed of a set of services, using Microsoft Windows Communication Foundation (WCF) Framework [1] for their endpoint to endpoint connection, that feed data about public transport for OPT's applications. This client-server model allows the computing partition of the tasks from the IMS modules and increases the server performance by dividing the workload of each module. This allows for any person to access services of the system through the World Wide Web, requiring only a client that needs to understand the application responses based on the established protocol. The usage of this kind of model also allows to include new modules in a easier way by developing them as independently from the other modules.

3.2.1 Logical Design

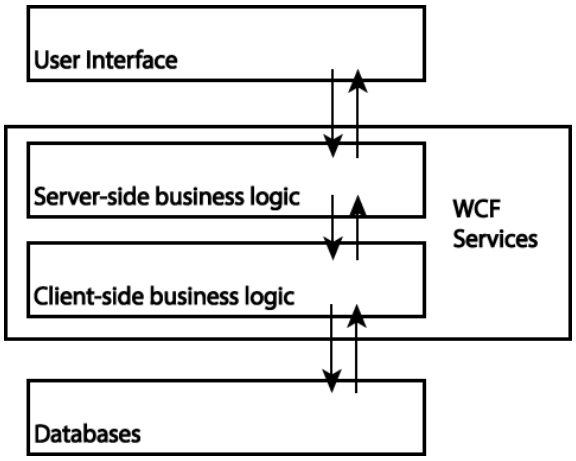


Figure 3.3: IMS Logical Architecture

Figure 3.3 illustrates the logical design of the Information for Mobility Support system. The first layer represents the user interface. This interface is either provided as a webapp, developed using ASP.NET MVC 2.5 Framework, or any of the mobile apps, Android and iOS based. The layer communicates with a second layer, the business logic layer, through multiple WCF Services implemented in C#, running on the .NET Framework 2.5. This second layer, besides the communication with the other layers, is also in charge with the network processing and the routing calculations.

The data used in this system is stored in an Oracle database and accessed using the OpenAccess Object Relation Mapper (ORM).

3.2.2 Conceptual Design

Figure 3.4 describes the conceptual design of the IMS system. The module *Proxy* is a module that manages all requests and routing inside the IMS system and, so, it can be called the central module.

[1] See http://msdn.microsoft.com/en-us/library/ms731082(v=vs.110).aspx for more details on WCF services

This is an important module because it establishes a frontier for outside connections, acting as point of access to the system. Around that connection management module there are others, which are not visible outside of the IMS network System. The *Monit*, *Mobile*, *WebClient* and *InfoBoard* are, respectively, the logger of the entire system, the module responsible for the connection to mobile devices of MOVE-ME, the module that generates the MOVE-ME webclient and a module specified for another software from OPT, the InfoBoard.

More related to the developed algorithm are the *PADA* and *BITA* modules. *PADA* is responsible for the algorithm that generates routes and returns the calculated trips. This module contains, at all times, a network graph in memory allowing a fast calculation of a fastest routes using the Dijkstra implemented using a priority queue to order the list of temporary paths between each iteration.

The *BITA* module it is the one that contains the layer that access the database where all the information of the physical network (operators, services, stops...) and also the temporal information of the network: the schedule of each service. These last two modules were the ones with the most interaction from the developments made as a part of this dissertation.

3.3 Provider Forecast Feeder

Altough not directly related with the thematics of the dissertation a tool, called Provider Forecast Feeder (PFF), was developed to understand the way the IMS system was implemented.

PFF is a service that was integrated inside the IMS architecture and allows providers who don't have any Real Time service to make immediate changes to the system. Since the IMS system was designed to pre-calculate a network (for three consecutive days) and keep it in cache for faster access, changes made by operators could only be made until three days before the inteded change to take effect - or, as an alternative, the system had to be restarted for changes to take place. This had many inconveniences like if the route of a bus service is partially on roadworks and there wasn't at least a 3-day notice, the providers weren't able to change their service schedule.

Although this service is not directly related with the thematics of the dissertation, its development allowed a strong knowledge of the IMS system architecture and its internal data structure.

PFF is, like the rest of the modules in the IMS system, a WCF service. This service communicates with BITA to receive, for every service of operators which don't have Real Time services, the schedule with the stops sequence and their respective transit time. All that data is stored in memory to allow a fast access and modification but, in order to prevent a big delay on starting the service if, for some reason, it went down, the data - whenever it finishes loading or a modification is made - is serialized and stored in files to persist.

Create, Remove, Update and Delete (CRUD) operations where implemented as part of PFF service's interface. In addition an operation to reset the changes made in a day and an operation do duplicate a trip with increment in its times - valuable for services with equal trips, through the same stops, during a day. To allow operators to modify to the loaded schedules, changes were made to the Portal generated by the WebAdmin module. An interface created with HTML5, CSS3 and

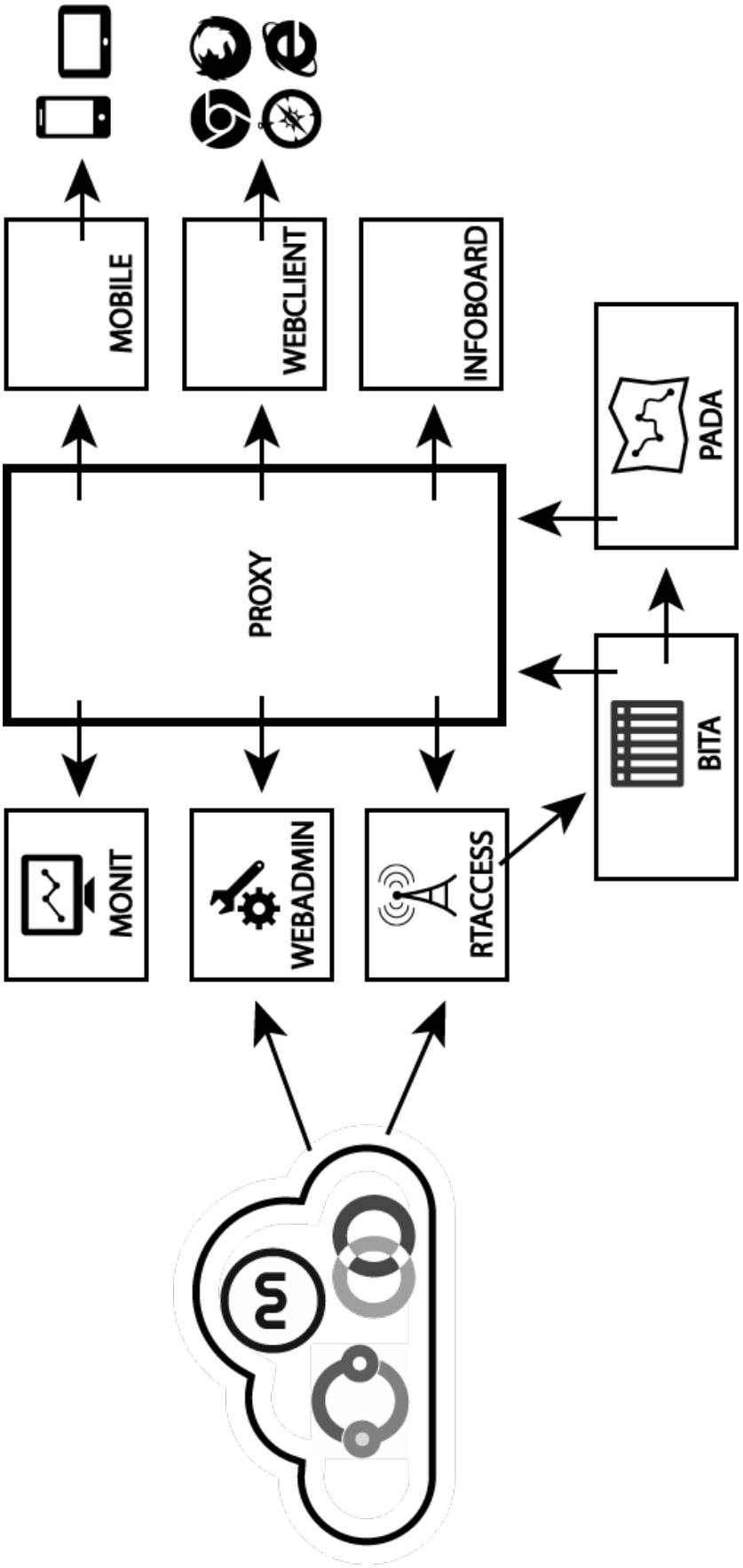


Figure 3.4: Information for Mobility Support core architecture conceptual model

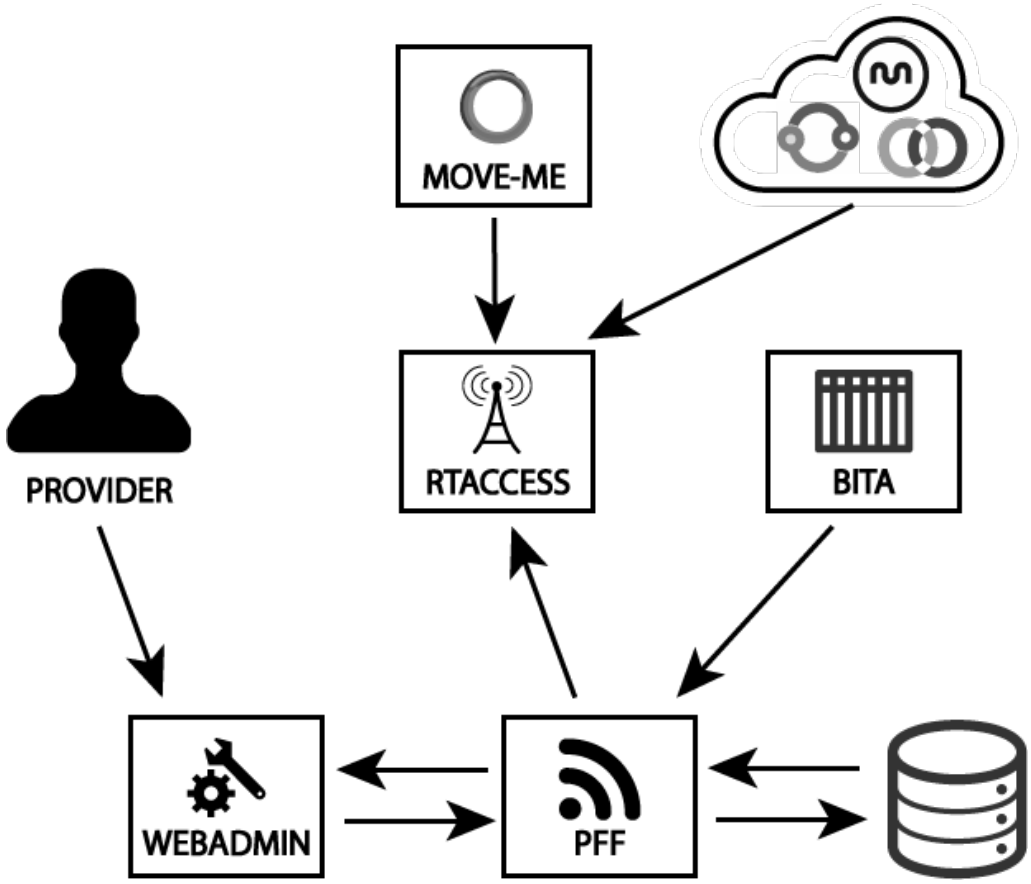


Figure 3.5: PFF interfaces with the IMS systems

jQuery was created to allow a fluid and interactive Schedules Editor for operators to change the information without much effort.

Chapter 4

Proposed Solution

The following chapter describes the proposed solution, a new module for the IMS system, and the idea to implement it. In order to validate the system as a working concept the scientific method as set of guiding principles was used. As Christin Wiedemann states "scientists want to find out how the world works; software testers want to know how the software they're testing works. Those two missions share a lot in common" [26]. To be able test the algorithm that was developed during the thesis, besides the validation with the Porto network, the following principles were followed:

- Create a model describing natural world;
- Use model to develop hypotheses;
- Run experiments to validate hypotheses.

The chapter contains the model describing a public transport network, the algorithm's hypothesis and its implementation in the IMS system. It also contains the expected results description the system is expected to return and its performance analysis.

4.1 Model of a Public Transport Network

Since OPT's internal structure of the graph network is similar to a $\mathbb{L}$ representation, as described on section 2.1.1.1, a scale-free network will be created to describe a real-word public transport network. On the researched literature, the Barabási-Albert Model arises as the most cited method to describe scale-free networks [27, 7]. The most notable characteristic in this kind of networks is the fact that it is common for some vertexes to have a degree that exceeds the average. These ones are often called the hubs. This property can be observed in public transport networks, where same stations are considered to be central and, so, they have many starting, ending or passing by services.

---

1 Output: scale−free multigraph

2 G = ({0,...,N−1}, E)

Table 4.1: Variables of the generated network

Variable	Value
No. Nodes	30
Min. Degree	2
No. Trips	10
Min. stops in a trip	3
Max. stops in a trip	10
Depth of each zone	2

```
M: array of length 2Nd
for (v=0,...,n−1)
    for (i=0,...,d−1)
        M[2(vd+i)] = v;
        r = random number from {0,..., 2(vd+i)};
        M[2(vd+i)+1] = M[r];
    end
end

E = {};
for (i=0,...,nd−1)
    E[i] = {M[2i], M[2i+1]}
end
```

Listing 4.1: BA Model pseudo-code implementation

Listing 4.1 contains the pseudo-code used for the implementation of Barabási-Albert Model for this network and table 4.1 details the values of the variables used to generate the test case.

The number of nodes is the total number of nodes contained in the network. The minimum degree is the minimum number of edges each node has. The number of trips is the number of generated trips for the test case, each a minimum of 3 stops and a maximum of 10. Depth of each zone is the depth the script for generating zones, a depth-first search algorithm. The degree distribution of this network, as a scale-free network, follow the power law $P(k) \sim k^{-\gamma}$.

Figure 4.1 shows a network generated using the previously mentioned algorithm. This network was visualized using vis.js [28], a browser-based graph visualization library.

As we can see, some nodes have a much larger degree then the average. Besides generating the network, fares, times of travel, zoning and trips of the nodes were also generated. For a better understanding and explanation of the proposed solution lets consider the following for this network:

- All edges are bi-directed;
- A trip may pass through the same node more than once but not by the same edge;
- Each edge on a trip acts as a `SubTrip` object (see section 4.2)
- Every stop inside has a connection to the other stops of the same zone, as observed in real world;

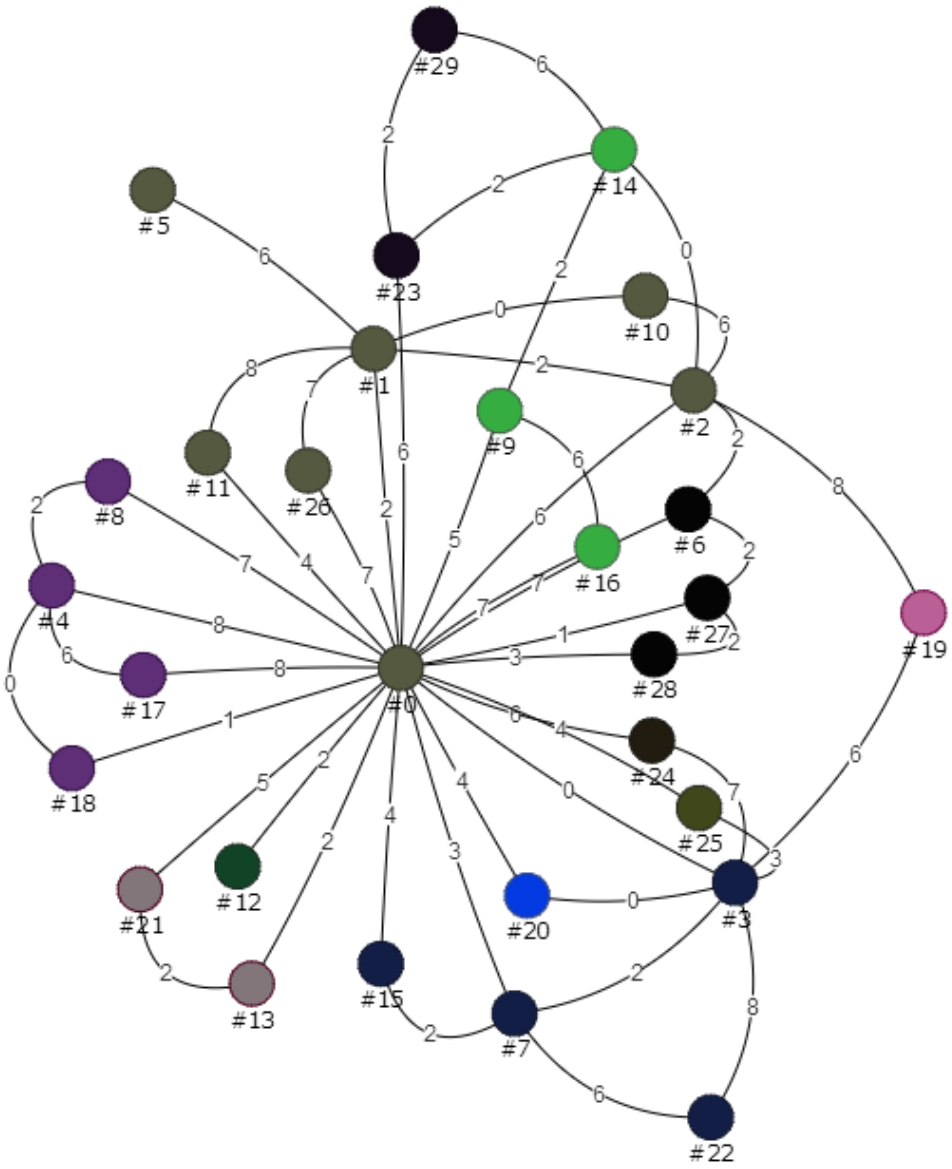


Figure 4.1: Randomly generated Public Transportation Network according to BA Model

In order to test multiple cases parametrized variables where defined for this process to be able to evaluate multiple cases. The values of these variables used on the test case shown on this document are described on table 4.1. Times of travel inside this network were randomly assigned between the minimum and maximum defined on the table. To generate the zones, the nodes where ordered from the one with the smallest degree and recursively the node and its neighbors were zoned following an depth-first search approach. Trips where generated by randomly picking a node then selecting random edges in order to create a path. The number of contained nodes was randomly picked between the minimum and maximum values set on the table.

The list of generated trips can bee seen on section 5.6 and the definition of the fares can be seen on the chapter of case studies, chapter 5.

4.2 Hypothesis

The algorithm proposed on this dissertation intents to calculate costs of trips on any network of public transports. The non-trivial aspect of this operation is the possibility of the existence of multiple fares on the same area of a network and even in the same trip which can lead to trips that are covered by more than one fare. On this last case, the proposed algorithm generates a list with information for the complete trip, with each necessary ticket and its corresponding price, place of first validation or purchase and segments of the trip the trip is valid for. This information is ordered by the total trip's price so the user knows his options to pay the cheapest price for a trip.

The input of the algorithm is `Trip` object which contains the begin and ending points of the trip, its total time and a list of the segments of the trip, the `SubTrip` object. `SubTrip` is an object from the IMS system that contains the passages through the network stops and other relevant information, which combined together give us a trip that can be either a simple bus trip or a combination of trips in multiple services and/or operators.

To calculate the cost of a trip the algorithm must already have been fed with information regarding the fares active on the chosen network. The are several different fare types that have been identified and were included in this algorithm and as it can be seen through analyzing the Unified Modeling Language (UML) specification of the Fare Calculator (see figure 4.5). According to the research made (see more at section 2.1.2) those fares were divided in the following categories:

Zoning

a kind of fare that has multiple zones, each containing a set of stops and a list of adjacent zones; the calculus of the cost of a trip is made by counting the number of crosses through the borders to another zone or to another ring (more on this systems can be found at section 2.1.2). An example of this kind of fare is the one implemented in the metropolitan area of Porto, the Andante (refer to section 2.1.2.1 for a more detailed description of this fare).

Tabled

this was the most common fare type found in the research made; most operators keep tables with the prices of a trip starting at stop A and ending at stop B. An example, in Portugal, of these kind of fare is the ones applied by regional transports operators like Rede Expressos and TransDev that have several stops spread across the country and the price of a trip between any of them is set by a table

Spacial

this kind of fare refers to a variable price according to the variable of the distance traveled on the trip.

Temporal

same kind of fare as the Spacial, but the variable that changes the price is the time spent on a trip.

While the costs of the first described category may be hard to calculate, because it may contain lots of variables, the other fare categories are easy to calculate as it consists in the access to information stored on tables of a mathematical calculation of a price according to a given variable (time or distance). With that in mind, one can say that the calculus of the cost of trip where only one fare is applied is a just a question of implementing an algorithm that calculates that price according to the trip's category.

However, when two or more fares coexist on the same trip, that calculation becomes trickier.

Figure 4.2: Sample trip where multiple fares co-exist

Let's image a trip between two points A and D with two transfers at B and C as illustrated on figure 4.2. For the purposes of the calculus of a fare the means of transport is irrelevant if the trip is made on a intermodal fare system, as described on section 2.1.2. Between A and B Fares 1 and 2 are applied and the trip's segment has a duration of 30 min, to the segment between B and C besides the previous fares the Fare 3 is also applied and it has a duration of 15 min and on segment between C and D only the Fares 2 and 3 are applied and the trip's segment has a duration of 50 min.

Intuitively a user may be complied to only choose Fare 2 for the trip since this is the only common fare on the entire trip. However this decision does not guarantee the cost of the trip to be the minimum possible.

The simplest case possible of fare configuration is if all of the fares are fixed and the tickets of each fare are only valid for one trip. If that happens, a greedy algorithm would be able to analyze the trip and return its minimum cost. Three more complex configurations are presented on tables 4.2, 4.3 and 4.4. The maximum duration of the ticket is set between the first entry on the service with that fare and must include the total time of the trip. Penalty exchange is the price paid to switch service.

Table 4.2: Fare Configuration no. 1

Fare	Price of service	Max. Duration	Penalty Exchange	No. Valid Services
Fare 1	2.50 €	∞	0.30 €	∞
Fare 2	3.00 €	∞	-	1
Fare 3	1.00 €	∞	-	1

With the previously presented configurations of fares a more complex analysis to find the cheapest price of the trip has to be made. To support the claim of which trip is the cheapest a decision tree has been created and it is presented on figure 4.3. Each end node of the figure has

Table 4.3: Fare Configuration no. 2

Fare	Price of service	Max. Duration	Penalty Exchange	No. Valid Services
Fare 1	1.50 €	∞	-	1
Fare 2	1.00 €	50 min	-	2
Fare 3	1.20 €	∞	-	1

a number that corresponds to a price on table 4.5 which presents all possible prices for the three described configurations of fares.

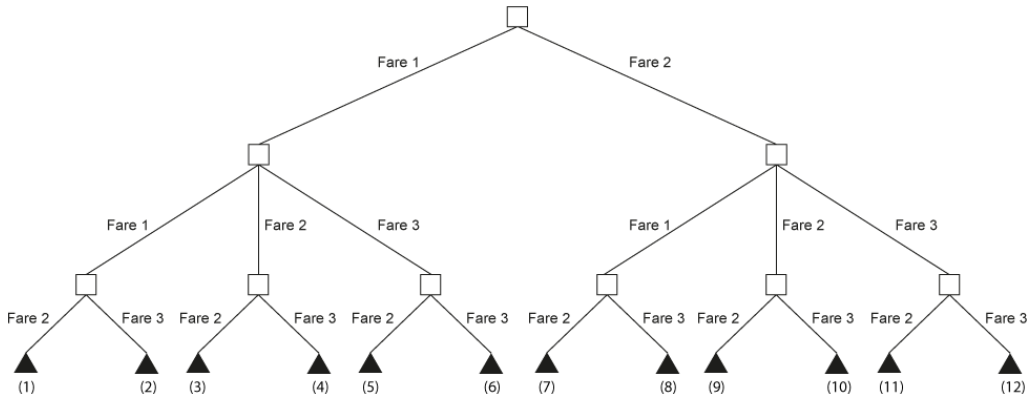


Figure 4.3: Decision tree to select fares a user should use to complete the trip illustrated on figure 4.2

Colored cells of table 4.5 represent the cheapest price for a given configuration. As it can be observed through its analysis each chosen sequence of fares can lead to a different price on the end node and different configurations on a network completely change the prices a trip may have.

When extrapolating this results to a real-world scenario where multiple fares can co-exist infinite possibilities of fare combinations may arise. With that in mind the algorithm to calculate a trip's cost implements a decision tree as the one on figure 4.3. After the algorithm calculate the results like the ones on table 4.5 they are sorted and a parameterizable limited number of results are returned to the user along with the choices required to be made along the trip.

4.3 Fare Calculator implementation

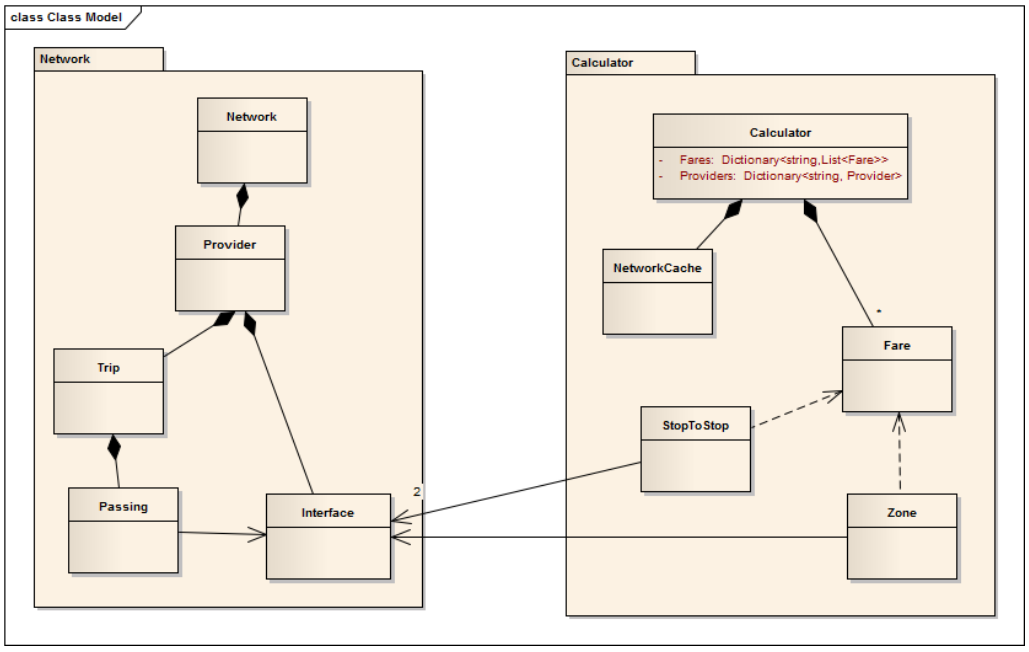
Fare Calculator, the developed system, is, like the rest of the IMS modules, implemented as a WCF Service. The system was developed to be integrated inside the IMS and allow a quick integration with the other modules, specially with MOVE-ME. The first part to be implemented on the Fare Calculator was the fare's datastructure.

In Figure 4.4 we can see a conceptual model that summarizes the calculator's interaction with the rest of the IMS system. The calculator has a main class, *Calculator*, that implements the WCF Service and loads the information stored in eXtensible Markup Language (XML) files into a cache. There are two XML files: one that contains every provider that will be available in the Fare

Table 4.4: Fare Configuration no. 3

Fare	Price of service	Max. Duration	Penalty Exchange	No. Valid Services
Fare 1	1.00 €	∞	0.50 €	∞
Fare 2	1.50 €	1 h 00 min	-	∞
Fare 3	0.50 €	∞	0.50 €	∞

Figure 4.4: Fare Calculator conceptual model of interactions with the IMS system



Calculator's system and its stops (or, in alternative, a flag that tells the parser inside *Calculator* to grab that info from the IMS system) and another that contains the fares to be added to the system. The XML Schema Definition (XSD) specification for those files is available at appendix A.

It is also on the `Calculator` class that the interface method *getTickets* is defined. This is the primary method to be called by the other IMS services. It returns a list of `PaymentPossibility` which is, no more, than an object with a list of `Ticket`'s and the total cost of the trip. This allows the system to reply more information than simply the price of the ticket. Each `Ticket` contains its own cost, the assigned fare, the time of the first validation, the corresponding first stop, the number of services where it will be used (or, in another words, the number of transfers) and a list with the `SubTrip` where it will be used With this returned structure an analogy with real life can be made where the user expects to receive a list with suggestions of payments that could be translated in natural language as:

> Your trip from A to C costs X €and you need a ticket one kind of fare from A to B and another different ticket at B to travel from B to C.

The Fare Calculator keeps two `Dictionary`[1]: one with an object `Provider` to store the

[1] http://msdn.microsoft.com/en-us/library/xfhwa508(v=vs.110).aspx

Table 4.5: Prices at each end node of the decision tree in figure 4.3 according to each fare configuration mentioned on tables 4.2, 4.3 and 4.4

Node / Config.	1	2	3	4	5	6	7	8	9	10	11	12
Config. 1	5.80€	3.80€	8.50€	8.50€	6.50€	4.50€	8.50€	6.50€	9.00€	7.00€	7.00€	5.00€
Config. 2	4.00€	4.20€	2.50€	3.70€	3.70€	3.90€	3.50€	3.70€	2.00€	2.20€	3.20€	3.40€
Config. 3	3.00€	2.00€	4.00€	3.00€	3.00€	2.00€	4.00€	3.00€	3.00€	2.00€	3.50€	2.50€

stops associated to each provider and another with the fares in the system. Figure 4.5 illustrates the UML specification of the internal structure `Fare`, used to store fares on the Fare Calculator cache.

Since there are several fares and each with their own characteristics the abstract factory design pattern [2] was used and the following abstractions of the superclass `Fare` were created:

1. Tabled
2. Space
3. Time
4. Zoning
5. FareSelector

`Tabled` is the superclass that holds the fares that are described in the form of a table: from point A to point B, the cost is X€. Its child classes are the `StopToStop` and `ZoneToZone` to store the prices of trips when fares are have a cost "from a stop to a stop" or have a cost "from a zone to a zone", for fares that contain zones with stops and the price is set according to the start and ending zones like the Transport for London (TfL) [29].

The `Space` and `Time` child classes of `Fare` are similar and used to store the variables for fares depending on the time spent on a trip or the distance traveled.

The `Zoning` child class is also an abstraction for the three types of existing zoning: `HoneyComb`, `Ring` and `Circular`. The calculus method for this three sub-divisions of zoning fares are the same but the decision of doing this division was to be able, in future, when retrieving information from the Fare Calculator to have the this knowledge to, for instance, be able to draw the zones and their corresponding stops.

The `FareSelector` class is a class the uses the composite design pattern [3] and it is an abstract class itself. The purpose of this implementation is to be able to add child classes of `FareSelector` that contain criteria for special cases of some fares, like an age criteria (some fares have discounts for the elderly, for instance) and a temporal criteria (some fares are active only partially during the day).

Each of this fares have some similar rules that are expressed on the `Rule` class. Those rules are

[2] http://sourcemaking.com/design_patterns/abstract_factory
[3] http://sourcemaking.com/design_patterns/composite

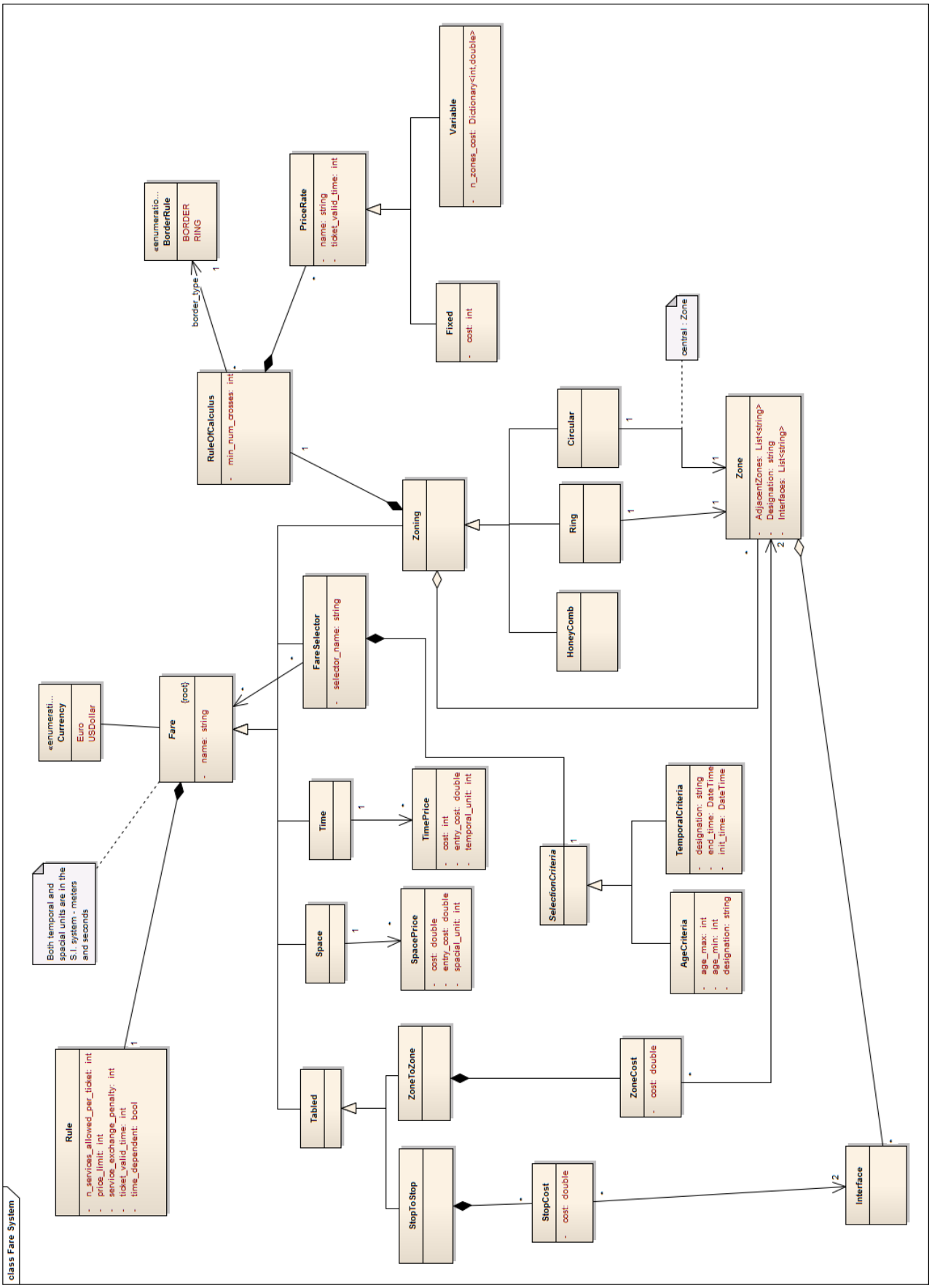


Figure 4.5: Fare Calculator class diagram UML specification

1. The price limit that a trip may have;
2. The extra price a user has to pay for at a transfer situation;
3. The maximum time the user ticket is valid;
4. The number of services allowed per ticket

For the zoning fares there is a class, `RuleOfCalculus`, that defines the kind of border rule for a zoning fare, the prices applied for each crossing of the frontier. Both the price and the valid time of a ticket can be either fixed or variable. The border rule for a zoning fare is a definition that references the two different methodologies of calculus:

1. The price is calculated by counting the number of border crosses made through the trip
2. The price charged for a title between two zones is calculated using the point of the beginning of the journey as the origin and incrementing that price whenever the user crosses the board for the next set of rings that surround that zone. This calculation applies recursively between the start and the finishing zone of the trip.

4.4 Performance Analysis

Since the Fare Calculator is a module to be included in the IMS system and to be integrated, primarily, in the MOVE-ME the time it takes to provide a solution is of critical importance. As it will be demonstrated on this section the time this operation takes is minimal, less than one second, and the solution is efficient for the defined purpose of being included into the sIMS.

To prove that the system performs well the time it took between receiving the `Trip` object with the information regarding a trip and the time it took for the algorithm to reply was measured. For the purposes of this analysis this time was called the “Decider Time”.

To do this test three networks where created using the method described on section 4.1. For each of this networks 50 trips where generated and requests where made to receive the prices of those trips. These trips where numbered between 0 and 49 and are represented on the x-axis on the charts presented on this section.

It was observed that the calculus of a single fare-trip between two points is immediate and so the complexity of the calculus made by the algorithm could only increase when the decision tree generated grows. That growth only occurs when a transfer happens. Since the theoretical trips generated for this performance analysis have the characteristics and the 3 fares used on test cases on chapter 5.1 each point of the trips generated can be considered as a point of transfer. This will also allow to test the efficiency of the code written to construct the tree.

The generated decision tree has a size equal to f^{n-1} and the number of end nodes is given by n^f where f is the number of fares and n the number of transfers made. This conclusion is possible because the generated tree is a perfect tree, which means the number children of each node is always the same except for the decision nodes.

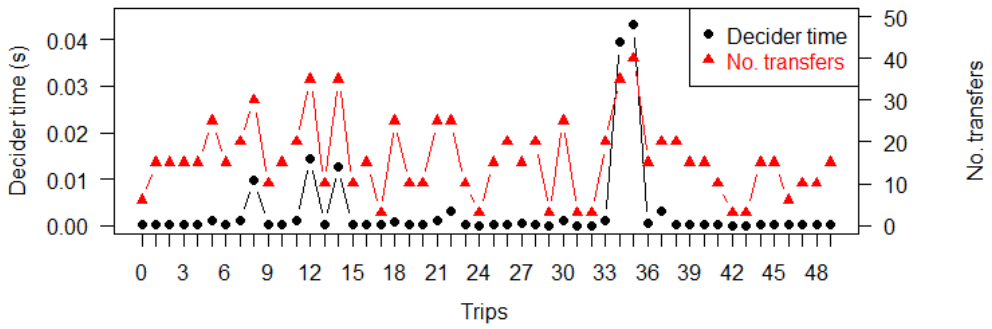


Figure 4.6: Chart comparing time spent on generating decision tree with a network composed of 300 nodes

Figure 4.6 represents the trips tested on a network composed of 300 nodes. This network's size represents approximately $\frac{1}{10}$ of the size of the physical network from Porto. The maximum "decider time" on this network was the trip no. 35 with a value of $4.330 * 10^{-3}$ s corresponding to a trip with 40 possible transfers. This generates a tree with a considerable size of $4.052 * 10^{18}$ and with 64000 end nodes.

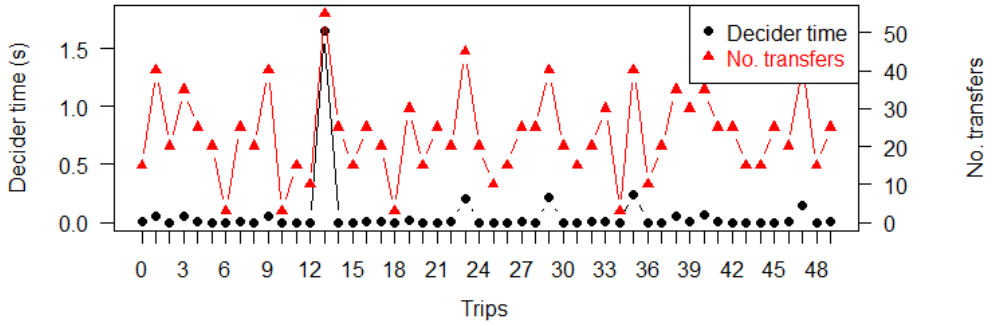


Figure 4.7: Chart comparing time spent on generating decision tree with a network composed of 3000 nodes

The chart represented on figure 4.7 shows the test values for the trips tested on a network composed of 3000 nodes, which is approximately a network with the size of the Porto's. On this network, the maximum "decider time" was of 1.651 s, for trip no. 13, corresponding to a trip with 55 possible transfers. This accounts for a tree size of $5.814 * 10^{25}$ and 166375 end nodes.

Figure 4.8 represents the trips tested on network with a 10000 nodes size. The maximum value of execution time for the "decider time" on this network was of 1.7334804s for trip no. 15 to which

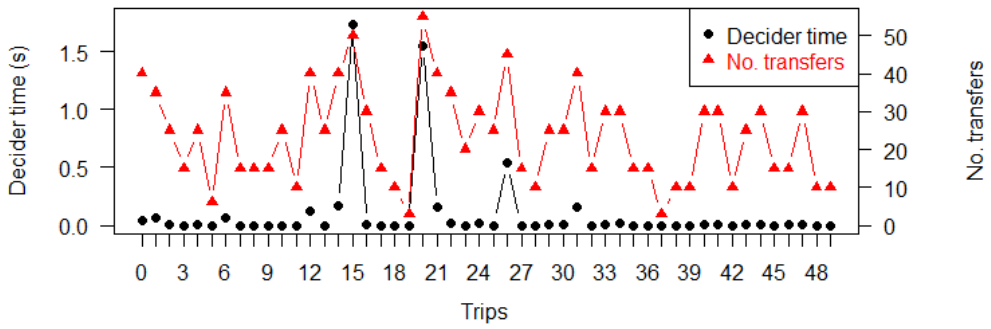


Figure 4.8: Chart comparing time spent on generating decision tree with a network composed of 10000 nodes

correspond 50 possible transfers. This values are where returned from a decision tree which had a size of $7.178 * 10^{23}$ nodes and had 125000 end nodes.

On this case, unlike expected, trip no. 20 has a higher amount of possible transfers (55) but its execution time was smaller, in the order of 1.5492268s. On this case the tree size and number of end nodes are the same as the worst-case scenario on the previously tested network, $5.814 * 10^{25}$ and 166375 respectively. This small and unexpected difference should be imputed to external factors of the computer executing the algorithm.

On average the algorithm took 0.05145s to return a result and the median time for that operation was of $8.5605 * 10^{-4}$ s. The mode of the times extracted from the tests was of $2.65 * 10^{-5}$ s, $3.46 * 10^{-5}$ s, $3.76 * 10^{-5}$ s, $8.42 * 10^{-5}$ s, $9.06 * 10^{-5}$ s,$1.094 * 10^{-4}$ s and $2.617 * 10^{-4}$ s.

From this temporal analysis some conclusions can be drawn.

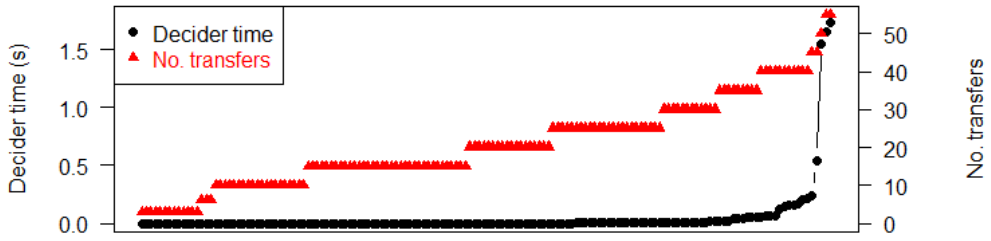


Figure 4.9: Chart comparing time spent on generating decision tree and the number of transfers

An immediate one is that since the measured times are very similar, as expected, the network

size does not affect the performance of the algorithm. Another remark drawn from the presented data is that the algorithm performs very well for a limited number of transfers. The chart represented on figure 4.9 combines all values from the three tested networks in a ordered way. As the analysis of the chart indicates, once the transfer number reaches 40 the "decider time" increases exponential. This conclusions are supported by the algorithm being unable to calculate trips over 65 transfers due to the limit imposed on the reply by the service, which is of 2 min and 30 sec.

Extrapolating this conclusions to a real-world scenario of a Public Transport Network it can be stated that in theory the algorithm can be applied for any Public Transport Network since it is not dependent on the size of the network and the number of transfers for accepted times of execution are very high in a real-world scenario, where it is completely unacceptable for the user to make 40 transfers no matter the distance traveled.

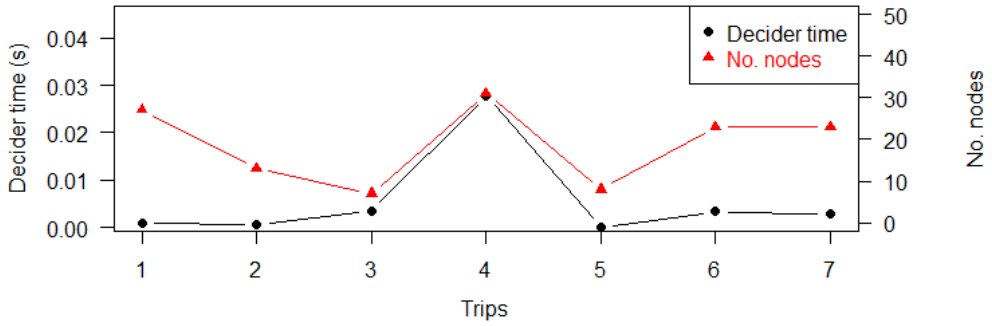


Figure 4.10: Chart comparing time spent on generating decision tree with the Porto Network

When doing the same tests using the test cases described on section for the Porto network, the results found are similar as shown on the chart from figure 4.10. The worst-case scenario has a value higher than the median presented on the theoretical test cases (0.027 759 6 s) for the trip between Aeroporto and Aveiro. This trip has 3 transfers and uses 3 different services but the number of nodes included on its path is low comparing to the theoretical cases. Since each trip is unique and has its unique fares there is no way of finding a relation between time and the number of nodes in a path.

As far as the OPT's expectations and the needs for the Public Transportation Network of Porto the algorithm performs well.

However the fact that the worst-case scenario being higher than the median calculated before on the theory cases, although not higher than the average, indicates that further tests on networks bigger than the Porto network are required. In spite of the algorithm being expected to present an efficient behavior on those networks, more complex trips found on those bigger networks are required to be tested before a concrete conclusion on the applicability of this algorithm for any trip inside any network is taken.

4.4.1 Route Builder option: cheapest trip

With a system able to calculate the cheapest price a user has to pay for a trip the intention of creating a new option on *PADA* to optimize the route builder emerged. This option allows the user to choose the cheapest trip when searching for an option. This option is different then the other existing optimizations since the cheapest trip is not necessarily the one that arrives first or that takes less time to complete.

To achieve this a simple change on the router builder have been made. The route builder algorithm implemented on IMS (Information for Mobility Support), the *PADA* module, is implemented using the Dijkstra algorithm implemented with priority queues for ordering the trips between the calculation. Since the price of a trip is a numeric value, in theory replacing the order method of the queue would produce the cheapest trip from point A to point B.

There is no preprocessing, or at least a simple implementation without changing the structure of the route builder, that allows to store in cache the values for every edge of each node since that the price of a trip changes according to its path. This means that for each iteration of the Dijkstra algorithm the price calculation of the trip, from the starting node to the point the algorithm is focusing, needs to be made. Because of the high number of edges expected on a public transport network that operation is not practicable.

A test was made to analyze the time it took for a trip to be built using this method to find the three cheapest trips between Aliados and Câmara de Gaia, a trip that the most simple outline is the Metro do Porto line D, where this stations are successive. As shown on table 4.6 this method is not a feasible for the calculator since it exceeds the maximum execution time for the route builder, which is of 60 s.

Table 4.6: Time to retrieve three trips between Aliados and Câmara de Gaia with the route builder optimized for price search

Trip	Time of Completion
#1	7468.989 ms
#2	20375.122 ms
#3	83421.341 ms

A solution for this cheapest trip option was proposed using results optimized by the trip duration and order them by the trips price. This solution produces an acceptable solution on the network from the IMS system.

Chapter 5

Case Study Results

To validate the obtained information from the Fare Calculator two Case Studies were defined:

Hypothetical Case Study uses a random network describing the real-word and trips with multiple transfers to prove the viability of the algorithm

Porto Case Study a Case Study that uses the data available from OPT of the public transportation network from the city of Porto

This chapter describes the variables on each case study and summarizes one of the results. The remaining results are interpreted similarly.

5.1 Hypothetical Case Study

5.1.1 Fares

To test the pricing of trips within an area, fares were created to be used as test cases. Tables 5.2, 5.3, 5.4 summarize the three fares used to test the given network.

Table 5.1: Fares used in Test Cases used to prove hypothesis

Table 5.2: Temporal Fare: Fake#1

Time (s)	Cost (€)
0	0.50
1	0.70
4	0.90
9	1.10
16	1.30
25	1.50
36	1.70
49	1.90
64	2.10
81	2.30

Table 5.3: Temporal Fare: Fake#2

Time (s)	Cost (€)
5	0.50
8	0.70
10	0.90
19	1.10
25	1.30
34	1.50
38	1.70
46	1.90
56	2.10
59	2.30

Table 5.4: Zoning Fare: Fake#3

# Zones	Cost (€)
1	0.80
2	1.00
3	1.20
4	1.40
5	1.60
6	1.80
7	2.00
8	2.20
9	2.40
10	2.60

Table 5.5: List of zones from fare Fake #3 and their respective stops and adjacent zones. Each zone is named "Z" and its identifying id.

Zone	**Adjacent Zone**	**Stops**
Z1	Z2, Z3, Z4, Z5, Z6, Z7, Z8, Z9, Z10, Z11	0, 1, 2, 5, 10, 11, 26
Z2	Z1	12,
Z3	Z1	4, 8, 17, 18,
Z4	Z1	13, 21,
Z5	Z1, Z7, Z8, Z9, Z10	3, 7, 15, 22,
Z6	Z1, Z12	9, 14, 16,
Z7	Z1, Z5	19
Z8	Z5, Z1	20,
Z9	Z1, Z5	24
Z10	Z5 Z1	6, 25
Z11	Z1	27, 28
Z12	Z1, Z6	23, 29

Both Fake#1 (table 5.2) and Fake#2 (table 5.3) are Temporal Fares. As referred in section 4.2 the price of trips covered by these is proportional to the time spent on the transport. The first column expresses time spent on inside a transport, in minutes for this test case, and the second one reflects the pricing in monetary units. Note that the table should be analyzed like, for example, "up to five minutes (inclusive) it costs 0.50 monetary units".

The times of the first fares were generated using the idea that since the maximum time of each edge is of 10min and the maximum edges that a trip can have is 10, the time was generated using the quadratic function $i * i$ where i is the iteration counter of the algorithm. The times of the second fare were generated using the function $random(last_inserd, last_inserted + 15)$ in order to create a fare with some randomness and that increases on steps between 0 and 15 minutes. On both cases the increment of the monetary unit (m.u.) was of 0.20. The reason for the different ways of generating these fares was to have multiple fares that are different but, at the same time, very similar since only some of the parameters vary.

The zoning fare, Fake#3 (table 5.4), was inspired on the Andante fare. The list of zones from this fare and their respective stops and adjacent zones is listed on table 5.5. Its zoning rule (see action 4.3) is, unlike the Andante's, a border rule which means that the number of zones crossed is the number of crosses made between borders. If a trip crosses the same border twice, two crosses are counted. More on this kind of fares can be found at section 2.1.2. The prices on this Fake#3 fare start at 0.80 monetary units and increase 0.20 for each type of ticket.

5.1.2 Trips

The trips used for this test case where randomly generated by choosing a node of the network and then selecting random edges to build a path. The number of nodes of the trip was randomly set between the values on table 4.1. The table presented bellow, table 5.6, lists the generated trips.

Table 5.6: Test Case trips

Trip Name	Cost
Trip #0	8 → 0 → 26 → 1 → 2 → 14 → 23
Trip #1	14 → 2 → 6 → 27
Trip #2	25 → 3 → 20
Trip #3	16 → 20 → 3 → 24 → 0 → 9 → 14 → 29
Trip #4	22 → 26 → 0 → 6 → 27 → 28 → 0 → 11

5.2 Case Study: Porto

In order to prove the feasibility of the algorithm created within this dissertation it was tested using the Porto Public Transport Network case study. IMS (Information for Mobility Support) system containing the Porto Network has the following characteristics:

- 11 207 sequences
- 3 728 sequences of traveling
- 7 479 sequences of walking
- 78 983 edges on network graph
- 3 032 stops (and, consequently 3032 nodes on network graph)
- 683 points of interest

As stated before on section 3.2.2 the PADA module is the part of the system responsible for searching and building routes for MOVE-ME using the information stored in BITA. The tests made on the Porto network are, therefore and as expected by the dissertation's proposal, made with trips already calculated and returned by PADA. The trips used as a Test Case were the following: →

1. FEUP → Castelo do Queijo
2. FEUP → Câmara de Gaia
3. São Bento → Espinho
4. Aeroporto → Aveiro
5. General Torres → Espinho
6. Póvoa de Varzim → Hospital de S. João
7. Hospital de S. João → Póvoa de Varzim

To achieve the path of the trip the RouteBuilder of PADA is used (more information about it can be found at section 3.2.2). To obtain a route from PADA a request is sent with the following (relevant) information: a list with places the trip must contain, time of start of the search, maximum time of end and a preference setting - either to optimize the trip's duration or the arrival time. Some of the obtained results from PADA contain multiple trips but only the one that takes less time to

complete was used to prove the algorithm's feasibility. For the trips listed above the time of start was the day 17/04/2014 between five and six o'clock in the afternoon, a five hour window for the maximum time to arrive at the destination and an optimization based on the trip's duration. Each obtained trip have different characteristics on the network. Table 5.7 summarizes those characteristics. The number of decision nodes of the generated tree is the number of possibilities of payment for the trip.

Table 5.7: Unique characteristics of each of the Test Case's trips

Trip	Characteristic	No. Providers	No. Fares	No. Decision Nodes
#1	This is a generic trip that uses STCP's services	1	1	1
#2	This is a trip that can be made by either STCP, Metro do Porto or the combination of both provider	2	1	1
#3	This is a trip that can be made by CP and has 2 active fares	1	2	2
#4	This is a trip that is composed by CP and Metro do Porto and it has segments where one fare is active and others where two are active	2	2	2
#5	This trip uses CP's services and has two different fares	1	2	2
#6	This is a known test case, where the direction of the trip affects its cost	1	1	1
#7	This is a known test case, where the direction of the trip affects its cost	1	1	1

5.3 Case Studies Results

The results the Fare Calculator returns are shown on appendix B for the first described case and on appendix C. For each trip on the Hypothetical Case Study 5 possibilities of payment are presented, since the randomness of the fares and the fact that at each node of the trip the user is able to transfer from service - creates creates a large number of possibilities.

This section will describe the example of Trip #0 from the Hypothetical Case Study and a small description of the case study from Porto.

Figure 5.1 illustrates the trip's path on the network the case study refers to. There are two prices that can be practiced over this trip according to the fares described on table 5.1 with different configurations for the tickets purchase order. This first and cheapest price, is a ticket from the Fake#HoneyComb fare which costs 1.40€.

The second possibility of payment for this trip is a possibility that uses two tickets: for the segments $8 \rightarrow 0 \rightarrow 26 \rightarrow 1 \rightarrow 2$, the Fake#HoneyComb fare is used; then, between $2 \rightarrow 14$ Fake Temporal#2 is used with a cost of 0.50 and between $14 \rightarrow 23$ the HoneyComb fare is used again. The second possibility has a cost of 1.70 m.u.

The third possibility uses the same tickets as the second but the Fake Temporal#2 fare is only used on the last segment of the trip.

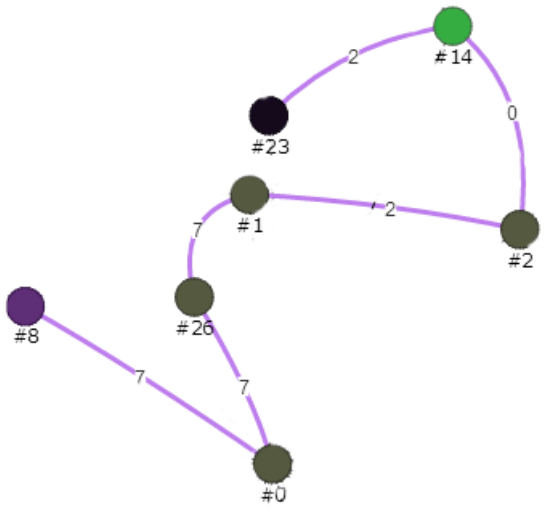


Figure 5.1: Trip 0 represented on Hypothetical Case Study network

The Porto Case Study results are dependent on the prices applied and the existent zones at the moment of test. Such information is available at appendix D. Each trip described for this Case Study contains the number of nodes search (which is equivalent to the number of stations the trip passes by), the total time it took for the solution to be requested, calculated and returned to a system and the time it took the decision tree to be constructed.

The results for each trip are presented the same way the previous Case Study is with each necessary ticket for a trip being shown. The ticket contains its price, fare, valid segments and the time of first validation or usage. The provider of the service for the trip is identifying in the valid segments, before the first stop of that provider, between squared brackets. If square brackets are found in the middle of a sequence of stops it represents a change of service/provider but not of fare.

Chapter 6

Discussion

The following section describes an alternative hypothesis that was studied during the dissertation and concludes with the validation strategy used and the authors global comments on the work done.

6.1 Alternative Hypothesis

To solve the problem of calculating calculating the price of a trip on a Public Transport Network two hypothesis were approached during the course of this dissertation.

Besides the hypothesis described on section 4.2 another methodology, which it will be called "Dijkstra Solution", was researched. This methodology to solve the problem was based on changing the routing algorithm system, PADA, that returns finds routes for the MOVE-ME. As the name suggests the approach would be done by using the Dijkstra algorithm.

On the "Dijkstra Solution" the solution for finding prices of fares based on the time variable, the spacial variable between two points would become trivial since it is the application of the Dijkstra problem. For fares based on tables it would be a matter of consulting the dictionaries that store such prices whenever the Dijkstra reaches a new node.

To implement fares based on zoning it would become trickier but still possible. Each node of the graph stored on PADA would have information about the zone it belongs and the links between two nodes of the graph where crossing of zoning border occurs would be flagged as such and so, with the same methodology used on the presented solution for zoning fares presented on the hypothesis section (4.2), at each iteration of the Dijkstra algorithm would be possible to find out the price of the trip so far since the zone of start and the zones the trip has passed by are known.

At the beginning of the routing PADA launches threads to analyze each possible solution of optimum path to the destination. Each of these threads would store the sequence of fares for the route they were calculating to be able, at any given point of the routing, to know the price of the journey so far.

This solutions appears to solve the problem of calculating the trip and finding the cheapest trip. However when confronted with edges where multiple fares are active this algorithm would be forced to launch a new thread to find out which of those would be the best solution. Since the

Dijkstra algorithm searches for the shortest path in every direction of the graph, the launching of a new thread on PADA could become a problem on a big networks with multiple fares since there is no way of predicting the number of threads being launched and its computational costs.

In addiction to the number of threads launched there is the computational cost of calculating the price of a trip, especially if it is a zoning fare. Although the cost of calculating the price of a trip for a thread shouldn't be high by itself, when summing the cost of every thread the Dijkstra launches that cost would increase and the risk of running out of memory for making the calculus would be very high.

Another reason for not pursuing this alternative hypothesis was the fact that the PADA system is very complex and the time it would take to study the changes necessary to implement this hypothesis wouldn't fit the duration of this dissertation and the objective of this work was to have a system ready to use.

6.2 Validation

To validate the system implemented onto the IMS there are three main requirements the system must fulfill:

- The prices the system calculates must be accurate;
- The time of researching a trip with MOVE-ME shouldn't be affected by the price calculus;
- The algorithm must scale well.

To validate the first requirement tests where made using trips on the Porto network. The trips stated on section 5.2 along with several trips that started and ended on different zones of the Andante system, circular trips that had its ending and starting point at the same node but had different paths and complex trips that were expected to be valid using a certain title of Andante but due to its duration higher and more expensive titles were required (see section D.1 of appendix D for more info) were used.

Since the titles required for using in those trips were correctly calculated the first parameter was validated.

The second requirement was completely meet as shown on the performance analysis section (4.4). The time the the calculus of prices takes is irrelevant and it doesn't affect the time a user of MOVE-ME waits when using the application to find a route.

As the third requirement although with the used dataset the algorithm having a good performance and the theoretical results indicate that the algorithm scales well a conclusion about larger networks cannot be provided before testing the algorithm with larger datasets where more multi-fare trips are found.

6.3 Global Comments

The developed calculator tool was satisfactory. Although the used dataset did not had many trips with multiple fares some test cases for new fare systems inside the given dataset were created to ensure the fare calculator tool worked properly with several fares.

Another satisfactory outcome of the produced work was the UML model that, along with the research done on fares, is expected to fit for any fare model in practice on any Public Transportation Network.

The solution for the problem of finding the less expensive trip for a user was not completely solved. The created system does no attempt on trying to find such solution. Instead, when integrating the calculus of fares with MOVE-ME, the both the fastest trips and the trips that arrive earlier at the destination are calculated and the fare calculator orders them by its price and returns them to the user. Since the main use case of MOVE-ME is for finding trips that start within a small temporal window this solution was accepted by OPT as valid.

The choice of implementation as an opposite to the alternative hypothesis described on this chapter has an advantage that is worth mentioning: if the system is intended to scale to, for example, the entire country of Portugal changes on this system are not expected to be made. This way OPT can continue the develop and increase of value of its systems.

Chapter 7

Final Remarks and Future Work

A global assessment of the project developed under this master thesis is presented on this chapter. A section about future work to improve the created system is also included.

The creation of the Fare Calculator system was a necessity at OPT to complement the information the users of MOVE-ME receive. Since the algorithm created was meant to be included in the Information for Mobility Support (IMS) Web stack there were a predefined set of rules for the development, a set of inputs (the networks of Porto and Lisbon) and expected outputs. In spite of the already defined set of inputs for the tool, the development had always in mind the possibility of the calculator to be used on different public transportation network, with different fares then the ones existing at OPT's current solution and with very large networks.

The research for fares on several public transportation networks - Porto, Lisbon, London, Amsterdam, Madrid [11, 13, 14, 30] - and the research about fares integration [12] allowed the creation of a data structure[1] for the system to allow the price calculation of any type of fare loaded into the system.

The results found on section 4.4 along with the previously mentioned data structure allow the conclusion that the Fare Calculator system should scale well since it is efficient and so it can be applied to larger situations then the available dataset.

7.1 Future Work

While the development of this project allowed the creation of a tool that fits the needs of the problem, many opportunities to extend the scope of the dissertation and expand the system remain.

The system is ready to accept any kind of fare the prices and tickets it returns are meant to be for users using occasional tickets. An improvement on the Fare Calculator system would be to allow the input of travel cards that reflect on the calculus of a trip. Another improve on the system should be to create methods that calculate the cheapest trip between two points on a network without using other pre-calculated results, as described on section 4.4.1. Combining the calculus of cheapest trips with the input of travel cards would allow the system to return to a user a trip based

[1] http://www.britannica.com/EBchecked/topic/152190/data-structure

on their, for instance, their monthly travel card that preferentially takes place on the covered areas of that travel card.

Another improvement that could be made to the algorithm would be the usage of pruning and branch and bound methods for the decision tree. Although the performing time of the algorithm being completely satisfactory, a performance boost should never be neglected.

Appendix A

XSD Definition of Fare Caculator input files

The following appendix contains the XSD Schema Definition for the two XML files used by the Fare Calculator to load data into its internal datastructure. The usage of this files is described on section 4.3 of this report.

Listing A.1: Fare file XSD Schema Definition

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="fares">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="fare">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="parameters">
                <xs:complexType>
                  <xs:attribute name="PriceLimit" type="xs:unsignedByte" use="required" />
                  <xs:attribute name="ServiceExchangePenalty" type="xs:unsignedByte" use="required" />
                  <xs:attribute name="TicketValidTime" type="xs:unsignedByte" use="required" />
                  <xs:attribute name="NServicesAllowedPerTicket" type="xs:unsignedByte" use="required" />
                  <xs:attribute name="TimeDependent" type="xs:string" use="required" />
                </xs:complexType>
              </xs:element>
              <xs:element name="providers">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element maxOccurs="unbounded" name="provider" type="xs:string" />
                  </xs:sequence>
```

```
22                  </xs:complexType>
23                </xs:element>
24                <xs:element minOccurs="0" name="space">
25                  <xs:complexType>
26                    <xs:sequence>
27                      <xs:element maxOccurs="unbounded" name="cost">
28                        <xs:complexType>
29                          <xs:attribute name="unit" type="xs:unsignedByte" use="required" />
30                          <xs:attribute name="price" type="xs:decimal" use="required" />
31                        </xs:complexType>
32                      </xs:element>
33                    </xs:sequence>
34                  </xs:complexType>
35                </xs:element>
36                <xs:element minOccurs="0" name="time">
37                  <xs:complexType>
38                    <xs:sequence>
39                      <xs:element maxOccurs="unbounded" name="cost">
40                        <xs:complexType>
41                          <xs:attribute name="unit" type="xs:unsignedByte" use="required" />
42                          <xs:attribute name="price" type="xs:decimal" use="required" />
43                        </xs:complexType>
44                      </xs:element>
45                    </xs:sequence>
46                  </xs:complexType>
47                </xs:element>
48                <xs:element minOccurs="0" name="table">
49                  <xs:complexType>
50                    <xs:sequence>
51                      <xs:element maxOccurs="unbounded" name="cost">
52                        <xs:complexType>
53                          <xs:attribute name="origin" type="xs:string" use="required" />
54                          <xs:attribute name="destination" type="xs:string" use="required" />
55                          <xs:attribute name="price" type="xs:decimal" use="required" />
56                        </xs:complexType>
57                      </xs:element>
58                    </xs:sequence>
59                  </xs:complexType>
60                </xs:element>
61                <xs:element minOccurs="0" name="parametersofcalculus">
62                  <xs:complexType>
63                    <xs:sequence>
64                      <xs:element name="rates">
65                        <xs:complexType>
66                          <xs:sequence>
```

```
67                      <xs:element maxOccurs="unbounded" name="rate">
68                        <xs:complexType>
69                          <xs:sequence>
70                            <xs:element maxOccurs="unbounded" name="cost">
71                              <xs:complexType>
72                                <xs:attribute name="n_zones" type="xs:unsignedByte" use="
                                      required" />
73                                <xs:attribute name="cost" type="xs:decimal" use="required" />
74                              </xs:complexType>
75                            </xs:element>
76                          </xs:sequence>
77                          <xs:attribute name="name" type="xs:string" use="required" />
78                          <xs:attribute name="maxtime" type="xs:unsignedShort" use="required"
                                />
79                          <xs:attribute name="rateType" type="xs:string" use="required" />
80                        </xs:complexType>
81                      </xs:element>
82                    </xs:sequence>
83                  </xs:complexType>
84                </xs:element>
85              </xs:sequence>
86              <xs:attribute name="borderRule" type="xs:string" use="required" />
87            </xs:complexType>
88          </xs:element>
89          <xs:element minOccurs="0" name="zones">
90            <xs:complexType>
91              <xs:sequence>
92                <xs:element maxOccurs="unbounded" name="zone">
93                  <xs:complexType>
94                    <xs:sequence>
95                      <xs:element name="designation" type="xs:string" />
96                      <xs:element name="adjacentzones">
97                        <xs:complexType>
98                          <xs:sequence>
99                            <xs:element maxOccurs="unbounded" name="zone">
100                             <xs:complexType>
101                               <xs:attribute name="name" type="xs:string" use="required" />
102                             </xs:complexType>
103                           </xs:element>
104                         </xs:sequence>
105                       </xs:complexType>
106                     </xs:element>
107                     <xs:element name="stops">
108                       <xs:complexType>
109                         <xs:sequence minOccurs="0">
```

```
110                                 <xs:element maxOccurs="unbounded" name="stop">
111                                   <xs:complexType>
112                                     <xs:attribute name="code" type="xs:string" use="required" />
113                                   </xs:complexType>
114                                 </xs:element>
115                               </xs:sequence>
116                             </xs:complexType>
117                           </xs:element>
118                         </xs:sequence>
119                       </xs:complexType>
120                     </xs:element>
121                   </xs:sequence>
122                 </xs:complexType>
123               </xs:element>
124             </xs:sequence>
125             <xs:attribute name="name" type="xs:string" use="required" />
126             <xs:attribute name="currency" type="xs:string" use="required" />
127             <xs:attribute name="type" type="xs:string" use="required" />
128           </xs:complexType>
129         </xs:element>
130       </xs:sequence>
131     </xs:complexType>
132   </xs:element>
133 </xs:schema>
```

Listing A.2: Providers file XSD Schema Definition

```
134 <xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www
        .w3.org/2001/XMLSchema">
135   <xs:element name="providers">
136     <xs:complexType>
137       <xs:sequence>
138         <xs:element maxOccurs="unbounded" name="provider">
139           <xs:complexType mixed="true">
140             <xs:sequence minOccurs="0">
141               <xs:element name="stops">
142                 <xs:complexType>
143                   <xs:sequence>
144                     <xs:element maxOccurs="unbounded" name="stop">
145                       <xs:complexType>
146                         <xs:sequence>
147                           <xs:element name="code" type="xs:string" />
148                           <xs:element name="name" type="xs:string" />
149                           <xs:element name="coordx" type="xs:unsignedByte" />
150                           <xs:element name="coordy" type="xs:unsignedByte" />
151                           <xs:element name="restriction" type="xs:unsignedByte" />
```

```
                        </xs:sequence>
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
            <xs:attribute name="name" type="xs:string" use="required" />
            <xs:attribute name="ims" type="xs:boolean" use="required" />
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Appendix B

Hypothetical Case Study Results

The contents on the pages that follow are the result produced by the Fare Calculator algorithm for the test case described on section 5.1. Each trip has 3 possibilities of payment displayed over three columns of text. To each possibility a number of tickets is available and its cost and valid segments of the trip are described. If between valid segments a line is left in blank it means that the trip using that ticket is not continuous.

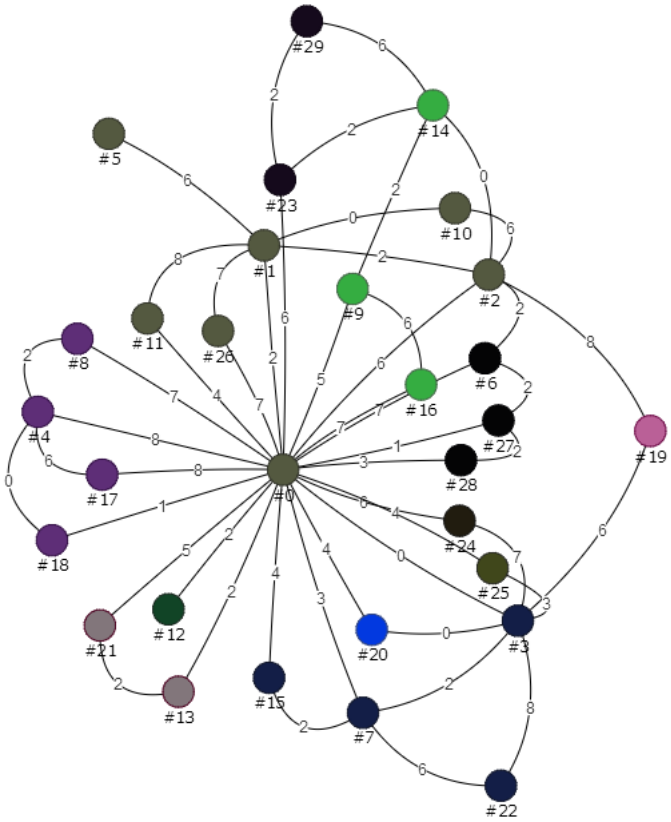


Figure B.1: Randomly generated Public Transportation Network according to BA Model

Figure B.1 illustrates the PTN used for the test cases represented on this appendix. For more information about this network see section 4.1.

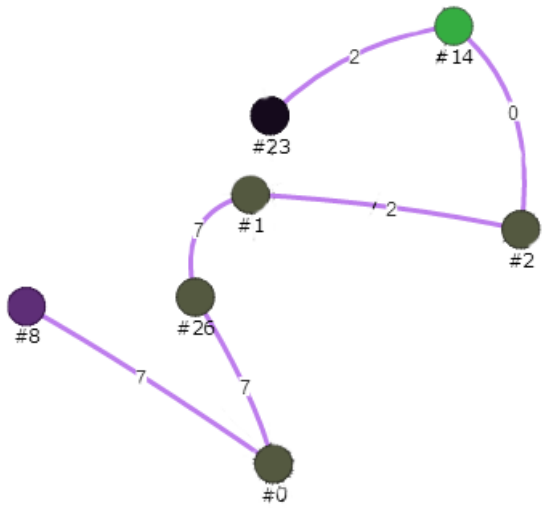


Figure B.2: Trip #0

Trip Name: Trip #0

Possibility #1
Cost of trip: 1,20
Number of tickets: 1

Ticket #1
Cost: 1,20
Fare: Fake#HoneyComb
Valid Segments:
8 → 0 → 26 → 1 → 2 → 14 → 23

Possibility #2
Cost of trip: 1,50
Number of tickets: 2

Ticket #1
Cost: 1,00
Fare: Fake#HoneyComb
Valid Segments:
8 → 0 → 26 → 1 → 2

14→ 23

Ticket #2
Cost: 0,50
Fare: Fake Temporal#2
Valid Segments:
2 → 14

Possibility #3
Cost of trip: 1,50
Number of tickets: 2

Ticket #1
Cost: 1,00
Fare: Fake#HoneyComb
Valid Segments:
8 → 0 → 26 → 1 → 2 → 14

Ticket #2
Cost: 0,50
Fare: Fake Temporal#2
Valid Segments:
14 → 23

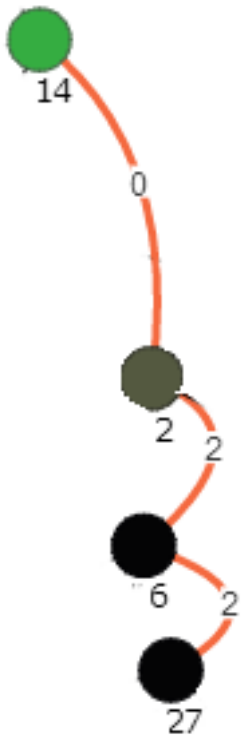


Figure B.3: Trip #1

Trip Name: Trip #1

Possibility #1
Cost of trip: 1,00
Number of tickets: 1

Ticket #1
Cost: 1,00
Fare: Fake#HoneyComb
Valid Segments:
14 → 2 → 6 → 27

Possibility #2
Cost of trip: 1,30
Number of tickets: 2

Ticket #1
Cost: 0,50
Fare: Fake Temporal#2
Valid Segments:
14 → 2

Ticket #2
Cost: 0,80
Fare: Fake#HoneyComb
Valid Segments:
2 → 6 → 27

Possibility #3
Cost of trip: 1,50
Number of tickets: 1

Ticket #1
Cost: 1,50
Fare: Fake Temporal#2
Valid Segments:
14 → 2 → 6 → 27

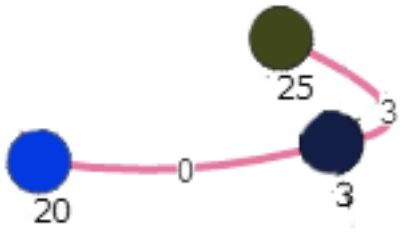


Figure B.4: Trip #2

Trip Name: Trip #2

Possibility #1
Cost of trip: 1,00
Number of tickets: 1

Ticket #1
Cost: 1,00
Fare: Fake Temporal#2
Valid Segments:
25 → 3 → 20

Possibility #2
Cost of trip: 1,20
Number of tickets: 1

Ticket #1
Cost: 1,20
Fare: Fake#HoneyComb
Valid Segments:
25 → 3 → 20

Possibility #3
Cost of trip: 1,20
Number of tickets: 2

Ticket #1
Cost: 0,50
Fare: Fake Temporal#2
Valid Segments:
25 → 3

Ticket #2
Cost: 0,70
Fare: Fake Temporal
Valid Segments:
3 → 20

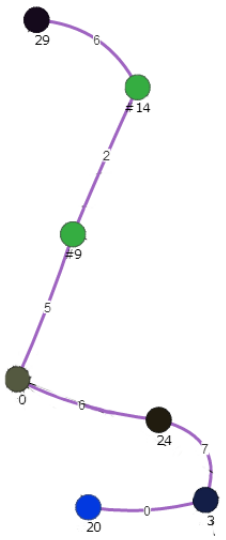


Figure B.5: Trip #3

Trip Name: Trip #3

Possibility #1	Possibility #2	Possibility #3
Cost of trip: 1,80	Cost of trip: 1,90	Cost of trip: 2,10
Number of tickets: 1	Number of tickets: 2	Number of tickets: 2
Ticket #1 Cost: 1,80 Fare: Fake#HoneyComb Valid Segments: 20 → 3 → 24 → 0 → 9 → 14 → 29	Ticket #1 Cost: 0,50 Fare: Fake Temporal#2 Valid Segments: 20 → 3	Ticket #1 Cost: 0,70 Fare: Fake Temporal Valid Segments: 20 → 3
	Ticket #2 Cost: 1,40 Fare: Fake#HoneyComb Valid Segments: 3 → 24 → 0 → 9 → 14 → 29	Ticket #2 Cost: 1,40 Fare: Fake#HoneyComb Valid Segments: 3 → 24 → 0 → 9 → 14 → 29

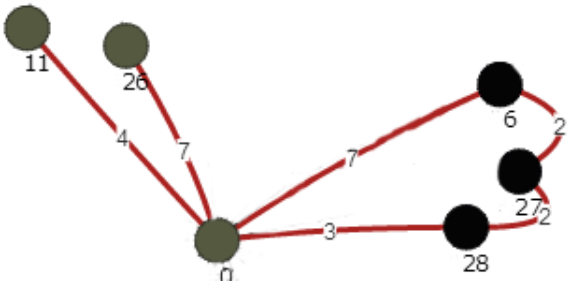


Figure B.6: Trip #4

Trip Name: Trip #4

Possibility #1	Possibility #2	Possibility #3
Cost of trip: 1,00	Cost of trip: 1,50	Cost of trip: 1,50
Number of tickets: 1	Number of tickets: 2	Number of tickets: 2
Ticket #1	Ticket #1	Ticket #1
Cost: 1,00	Cost: 1,00	Cost: 1,00
Fare: Fake#HoneyComb	Fare: Fake#HoneyComb	Fare: Fake#HoneyComb
Valid Segments:	Valid Segments:	Valid Segments:
26 → 0 → 6 → 27 → 28 → 0 → 11	26 → 0 → 6 → 27	26 → 0 → 6 → 27 → 28
	28 → 0 → 11	0 → 11
	Ticket #2	Ticket #2
	Cost: 0,50	Cost: 0,50
	Fare: Fake Temporal#2	Fare: Fake Temporal#2
	Valid Segments:	Valid Segments:
	27 → 28	28 → 0

Appendix C

Porto Case Study Results

The contents on the pages that follow are the result produced by the Fare Calculator algorithm for the Porto Case Study. Trips where created using the Route Builder of the *PADA* system as explained on section 5.2. The route builder may return several trips, with different paths, but only one was chosen - the one with less duration.

An explanation of the results presented here can be found at section 5.3.

Trip Name Hosp. S. João (Circunvalação) [STCP_HSJ12] - Castelo do Queijo [STCP_CQ9]

No. of Nodes 27

Total time of calculus 0.0940113s

Decision tree calculus time 0.0010163s

Possibilities of payment 1

Possibility #1

Cost of trip 1,20€

Number of tickets 1

Ticket #1

Cost 1,20€

Fare Andante

Time of first validation 17-06-14 16:33:00

Valid Segments [STCP] Hosp. S. João (Circunvalação)→ IPO (Circunval.) → S. Tomé → Amial → Capuchinhos → Via Norte (Circ.) → Mte Burgos (Circ.) → Congostas → Br. Sto. Eugénio → Quartel Militar → Alto do Viso → R. do Senhor → S.Ra Penhan→ Ruela → Rot. A.E.P. → Preciosa → Azenha de Cima

→ Lidador /Hospital → Quinta Ingleses → Real → Teatro Vilarinha → Parque da Cidade → D.Afonso Henriques → Dr. Afonso Cordeiro → Pr. Cid. Salvador → Edif. Transparente → Castelo do Queijo

---

Trip Name Hospital de São João [Hospital de São João] - Câmara de Gaia [Câmara de Gaia]

No. of Nodes 13

Total time of calculus 0.1473134s

Decision tree calculus time 0.0005168s

Possibilities of payment 1

Possibility #1

Cost of trip 1,50€

Number of tickets 1

Ticket #1

Cost 1,50€

Fare Andante

Time of first validation 17-06-14 16:26:00

Valid Segments [Metro do Porto] Hospital de São João → IPO → Pólo Universitário → Salgueiros → Combatentes → Marquês → Faria Guimarães → Trindade Inferior → Aliados → São Bento → Jardim do Morro → General Torres → Câmara de Gaia

---

Trip Name Porto (São Bento) [CP_Porto (São Bento)] - Espinho [CP_Espinho]

No. of Nodes 7

Total time of calculus 0.3466571s

Decision tree calculus time 0.0035024s

Possibilities of payment 2

Possibility #1

Cost of trip 1,70€

Number of tickets 1

Ticket #1

Cost 1,70€

Fare CP Monomodal

Time of first validation 17-06-14 16:05:00

Valid Segments [CP Linha Aveiro] Porto (São Bento) → Porto (Campanhã) → Gaia (General Torres) → Gaia (Devesas) → Valadares → Granja → Espinho

Possibility #2

Cost of trip 1,85€

Number of tickets 1

Ticket #1

Cost 1,85€

Fare Andante

Time of first validation 17-06-14 16:05:00

Valid Segments [CP Linha Aveiro] Porto (São Bento) → Porto (Campanhã) → Gaia (General Torres) → Gaia (Devesas) → Valadares → Granja → Espinho

---

Trip Name Zona Industrial [STCP_ZIND4] - Aveiro [CP_Aveiro]

No. of Nodes 31

Total time of calculus 0.2650473s

Decision tree calculus time 0.0277596s

Possibilities of payment 1

Possibility #1

Cost of trip 5,70€

Number of tickets 2

Ticket #1

Cost 2,30€

Fare Andante

Time of first validation 17-06-14 17:08:00«

Valid Segments Zona Industrial → Rua da Estrada → Crestins (Metro) → [Metro do Porto] Crestins → Esposade → Custóias → Fonte do Cuco → Senhora da Hora → Sete Bicas → Viso → Ramalde → Francos → Casa da Música → Carolina Michäelis → Lapa → Trindade → Trindade Inferior → Aliados → São Bento → Jardim do Morro → General Torres

Ticket #2

Cost 3,40€

Fare CP Monomodal

Time of first validation 17-06-14 18:00:00

Valid Segments [CP Linha Aveiro] Gaia (General Torres) → Gaia (Devesas) → Valadares → Espinho → Esmoriz → Ovar → Avanca → Estarreja → Cacia → Aveiro

---

Trip Name Gaia (General Torres) [CP_Gaia (General Torres)] - Espinho [CP_Espinho]

No. of Nodes 8

Total time of calculus 0.2183623s

Decision tree calculus time 0.0000050s

Possibilities of payment 2

Possibility #1

Cost of trip 1,50€

Number of tickets 1

Ticket #1

Cost 1,50€

Fare Andante

Time of first validation 17-06-14 17:00:00

Valid Segments [CP Linha Aveiro] Gaia (General Torres) → Gaia (Devesas) → Valadares → Espinho

Possibility #2

Cost of trip 1,70€

Number of tickets 1

Ticket #1

Cost 1,70€

Fare CP Monomodal

Time of first validation 17-06-14 17:00:00

Valid Segments [CP Linha Aveiro] Gaia (General Torres) → Gaia (Devesas) → Valadares → Espinho

---

Trip Name Póvoa de Varzim [Póvoa de Varzim] - Hospital de São João [Hospital de São João]

No. of Nodes 23

Total time of calculus 0.0502862

Decision tree calculus time 0.0027620s

Possibilities of payment 1

Possibility #1

Cost of trip 2,7

Number of tickets 1

Ticket #1

Cost 2,70€

Fare Andante

Time of first validation 06-17-14 17:44:00

Valid Segments [Metro do Porto] Póvoa de Varzim → Portas Fronhas → Vila do Conde → Varziela → Mindelo → Pedras Rubras → Senhora da Hora → Sete Bicas → Viso → Ramalde → Francos → Casa da Música → Carolina Michäelis → Lap a → Trindade → Trindade Inferior → Faria Guimarães → Marquês → Combatentes → Salgueiros → Pólo Universitário → IPO → Hospital de São João

---

Trip Name Hospital de São João [Hospital de São João] - Póvoa de Varzim [Póvoa de Varzim]

No. of Nodes 23

Total time of calculus 0.0534472s

Decision tree calculus time 0.0011085

Possibilities of payment 1

Possibility #1

Cost of trip 2,30€

Number of tickets 1

Ticket #1

Cost 2,30€

Fare Andante

Time of first validation 17-06-14 17:08:00

Valid Segments [Metro do Porto] Hospital de São João → IPO → Pólo Universitário → Salgueiros → Combatentes → Marquês → Faria Guimarães → Trindade Inferior → Trindade → Lapa → Carolina Michäelis → Casa da Música → Francos → Ramalde → Viso → Sete Bicas → Senhora da Hora → Pedras Rubras → Mindelo → Varziela → Vila do Conde → Portas Fronhas → Póvoa de Varzim

Appendix D

Fares from Porto Metropolitan Area

This appendix summarizes the key aspects of the fares that operators inside the IMS system use for their services. The information contained will also serve as support to the contents of the chapters on this dissertation.

D.1 Andante

Title Name	Price (€)	Max. Duration
Z2	1.20	1h 00m
Z3	1.50	1h 00m
Z4	1.85	1h 15m
Z5	2.30	1h 30m
Z6	2.70	1h 45m
Z7	3.05	2h 00m
Z8	3.45	2h 15m
Z9	3.80	2h 30m
Z10	4.20	2h 45m
Z11	4.60	3h 00m
Z12	5.00	3h 15m

Table D.1: Price and maximum duration of trips for each title of the Andante fare

The Andante is fare system active in the metropolitan area of Porto that works as an integrated ticketing system [4]. Table D.1 lists the prices of each title, according to the number of crossed zones, and to the maximum duration the title has. If the trip duration exceeds that maximum duration, the next title should be used. Maps with the zones of the Andante are shown on figures D.1 and D.2.

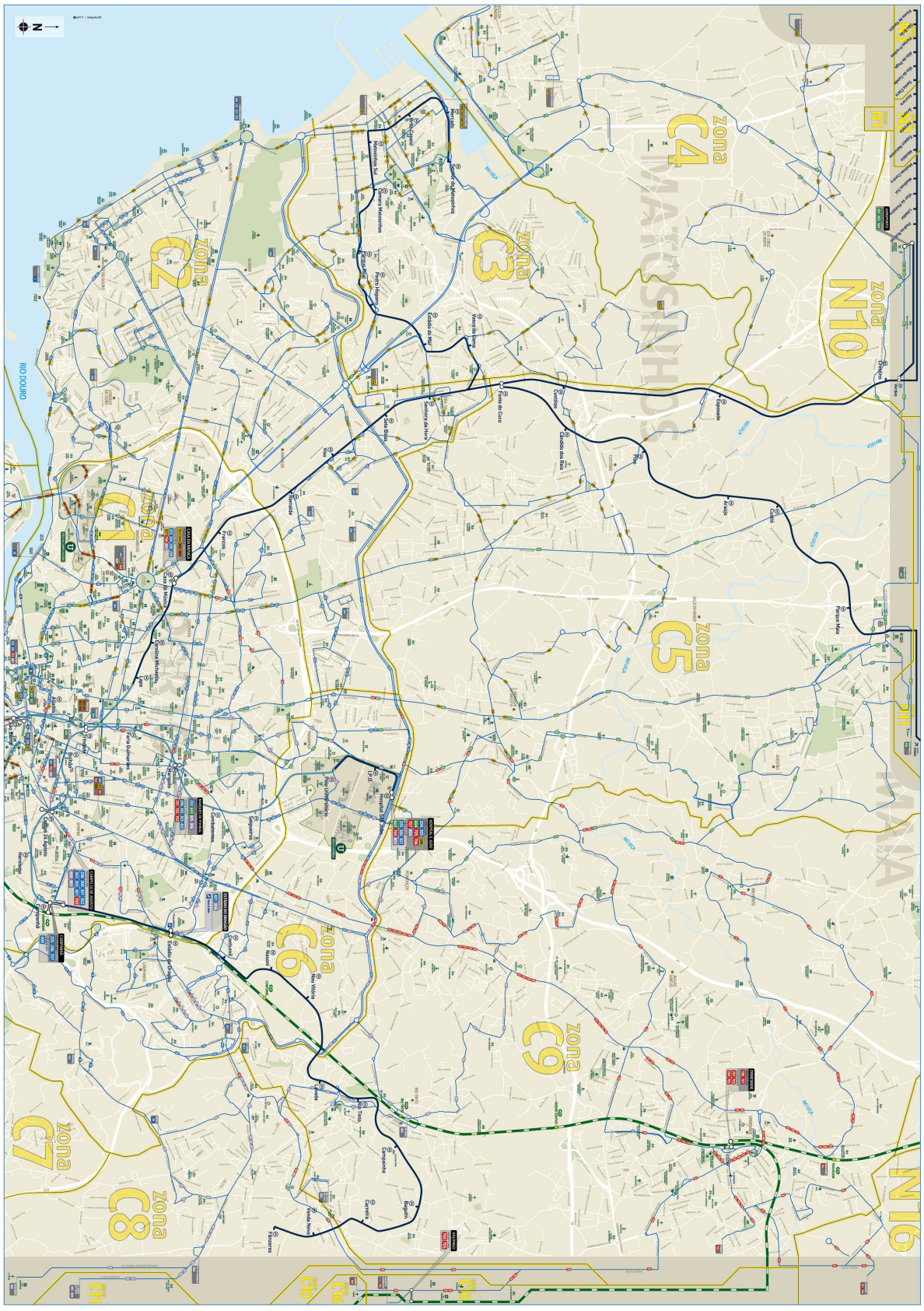


Figure D.1: Map of the zones from Andante in Porto metropolitan area - north zone

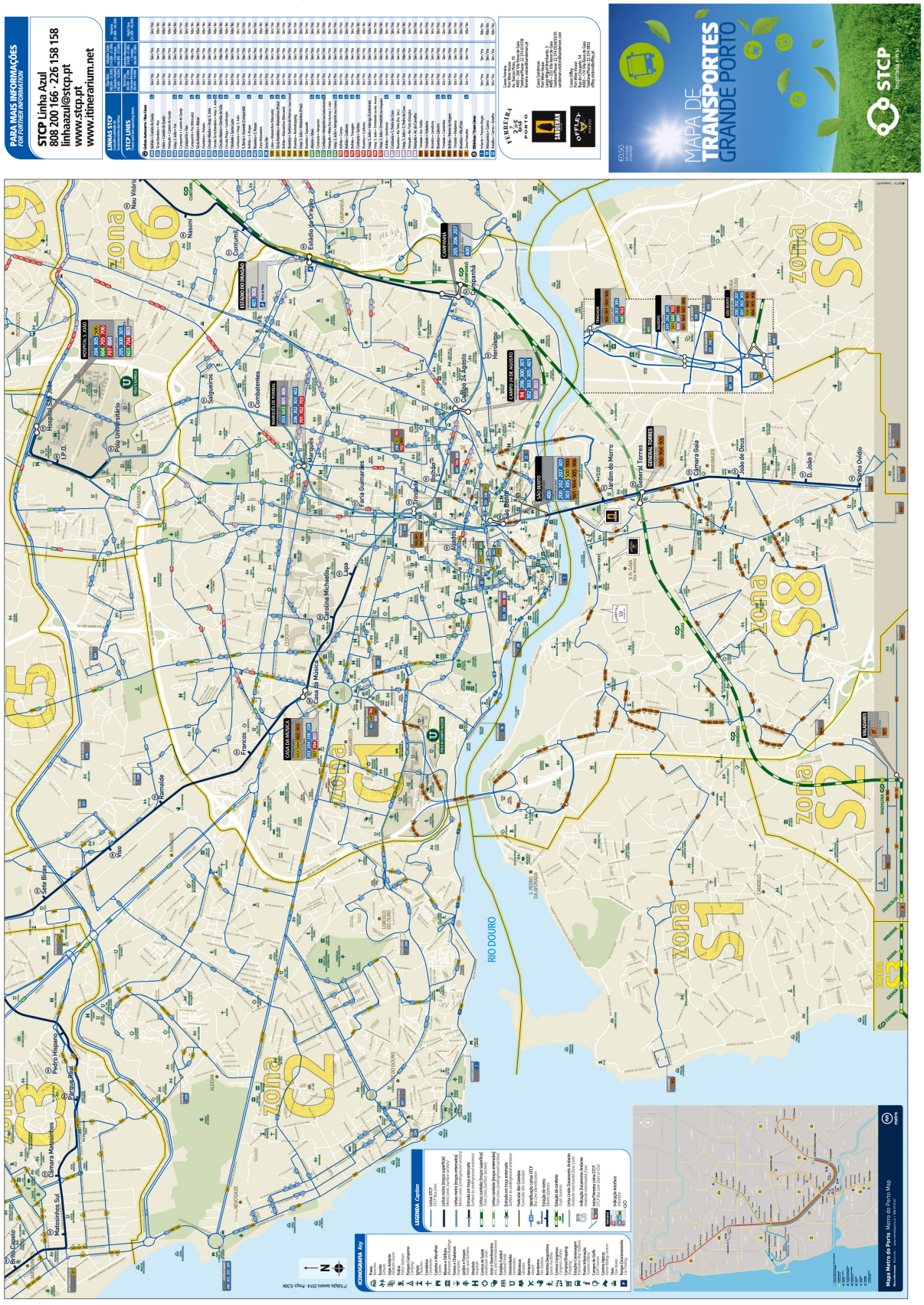


Figure D.2: Map of the zones from Andante in Porto metropolitan area - south zone

D.2 CP - Comboios Urbanos do Porto

"Comboios Urbanos do Porto" (translated as Urban Trains of Porto) is a set of train lines operated by CP - Comboios de Portugal - between Porto and other cities on its surroundings. The special case of this line is that it is partially covered by two different fares - for instance the Andante fare between Campanhã and Espinho and a fare tabled by CP called "CP Monomodal".

Both fares are shown, for described, on the next two pages.

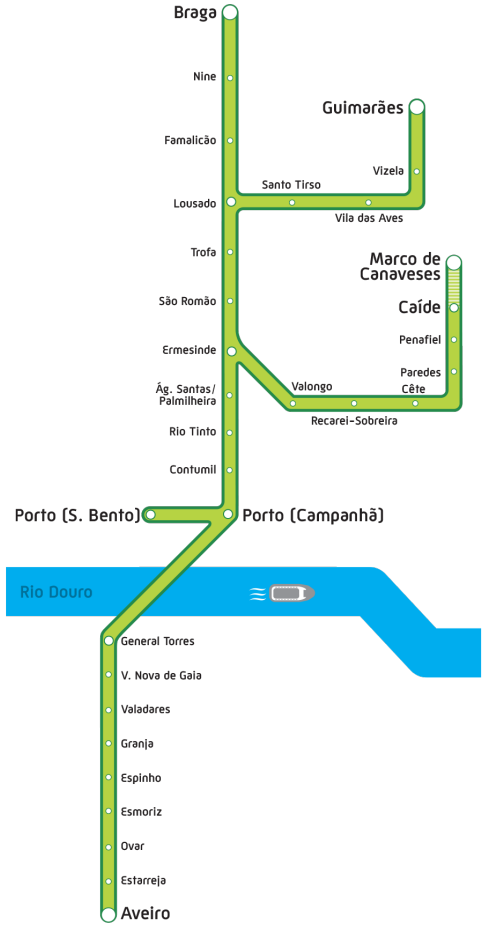
Prices of the zoning values of "CP Monomodal" for the regular ticket are described on table D.2.

Zone	Price (€)
1 and 2	1.40
3	1.70
4	1.95
5	2.25
6	2.55
7	2.85
8	3.10
9	3.40

Table D.2: Price of each zone for "CP Monomodal" regular ticket. The zones corresponding to each trip, given and entry and exit point, are described on the next page.

COMBOIOS URBANOS > PORTO

Mapa de Rede



Zonas de Percurso Tarifas Urbanas

Nota Importante:
Todos os títulos são válidos para um mínimo de 2 zonas.

	Porto (S. Bento)
Porto (Campanhã)	1
Contumil	1 1
Rio Tinto	1 1 1
Águas Santas / Palmilheira	1 1 1 2
Ermesinde	1 1 1 2 2
Travagem	1 1 1 1 2 2
Leandro	1 1 1 1 2 2 2
São Frutuoso	1 1 1 1 2 2 2 3
São Romão	1 1 1 1 2 2 2 3 3
Portela	1 1 1 2 2 2 2 3 3 3
Trofa	1 1 2 2 2 2 3 3 3 4 4
Lousado	1 1 2 2 2 3 3 3 3 4 4 4
Esmeriz	1 1 2 2 3 3 3 3 3 4 4 4 5
Barrimau	1 1 1 2 2 3 3 3 3 4 4 4 5 5
Famalicão	1 1 1 2 2 3 3 3 3 4 4 4 4 5 5
Mouquim	1 1 1 2 2 3 3 3 4 4 4 4 5 5 5 6
Louro	1 1 1 2 2 2 3 3 4 4 4 4 5 5 5 5 6
Nine	1 1 1 2 2 2 3 3 4 4 4 4 5 5 5 5 6 6
C. Cambeses	1 1 1 2 2 2 3 3 4 4 4 5 5 5 5 5 6 6 6
Arentim	1 1 1 2 2 2 3 3 3 4 4 5 5 5 5 6 6 6 6 7
Ruilhe	1 1 1 2 2 2 3 3 3 4 4 5 5 5 5 5 6 6 6 7 7
Tadim	1 1 1 2 2 2 3 3 3 3 4 4 5 5 5 6 6 6 6 7 7 7
Aveleda	1 1 1 1 2 2 2 3 3 3 4 4 5 5 5 6 6 6 6 6 7 7 7
Mazagão	1 1 1 1 2 2 2 3 3 3 3 4 4 5 5 5 6 6 6 6 7 7 7 8
Ferreiros	1 1 1 1 2 2 2 3 3 3 3 4 4 4 5 5 6 6 6 6 7 7 7 7 8
Braga	1 1 1 1 2 2 2 2 3 3 3 4 4 4 5 5 6 6 6 6 7 7 7 7 8 8

Porto (S. Bento)	
1	Porto (Campanhã)
1 1	Contumil
1 1 1	Rio Tinto
2 1 1 1	Águas Santas / Palmilheira
2 2 1 1 1	Ermesinde
2 2 1 1 1 1	Travagem
2 2 2 1 1 1 1	Leandro
3 2 2 2 1 1 1 1	São Frutuoso
3 3 2 2 2 1 1 1 1	São Romão
3 3 3 2 2 2 2 1 1 1	Portela
4 4 3 3 3 2 2 2 2 1 1	Trofa
4 4 4 3 3 3 3 2 2 2 1 1	Lousado
5 5 4 4 4 4 3 3 3 3 2 2 1	Stº. Tirso
6 5 5 5 4 4 4 4 3 3 3 2 2 1	Caniços
6 6 5 5 5 4 4 4 4 4 3 3 2 1 1	V. das Aves
6 6 6 5 5 5 5 4 4 4 3 3 2 2 1 1	Giesteira
7 6 6 6 5 5 5 5 5 4 4 3 3 2 2 1 1	Lordelo
7 7 6 6 6 5 5 5 5 4 4 3 3 2 2 1 1 1	Cuca
7 7 6 6 6 6 5 5 5 5 4 4 3 3 2 2 1 1 1	Pereirinhas
7 7 7 6 6 6 6 5 5 5 4 4 4 3 2 2 2 1 1 1	Vizela
8 7 7 7 6 6 6 6 6 5 5 4 4 3 3 2 2 2 1 1 1	Nespereira
8 8 7 7 7 7 6 6 6 6 5 5 4 4 3 3 2 2 2 1 1 1	Covas
8 8 8 7 7 7 7 6 6 6 6 5 5 4 3 3 3 2 2 2 2 1 1	**Guimarães**

Marco C.																							
Livração	1	1	1	2	2	2	3	3	3	4	4	4	4	5	5	5	6	6	6	7	7	7	8	8	8
Recesinhos	1	1	1	2	2	2	3	3	3	4	4	4	4	5	5	5	6	6	6	7	7	7	8	8	8
Vila Meã	1	1	1	2	2	2	3	3	3	3	4	4	4	4	5	5	5	6	6	6	7	7	7	8	
Oliveira	1	1	2	2	2	3	3	3	3	4	4	4	4	5	5	5	6	6	6	6	7	7	7		
Caíde	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	6	6	6	6	7	7				
Meinedo	1	1	2	2	2	2	2	3	3	3	4	4	4	5	5	5	6	6	6	7					
Bustelo	1	1	2	2	2	2	3	3	3	4	4	4	5	5	5	5	6	6	6						
Penafiel	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	5	6	6							
Paredes	1	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	6								
Oleiros	1	1	1	2	2	2	2	3	3	4	4	4	4	5	5	5									
Irivo	1	1	1	1	2	2	2	3	3	3	4	4	4	4	5	5									
Cête	1	1	1	1	2	2	2	3	3	3	4	4	4	4	5	5									
Parada	1	1	1	1	2	2	2	3	3	3	4	4	4	4	5										
Recarei-Sobreira	1	1	1	2	2	2	3	3	3	4	4	4													
Trancoso	1	1	1	1	2	2	3	3	3	3	4	4													
Terronhas	1	1	1	1	2	2	3	3	3	3	3	4													
S. Mart. Campo	1	1	2	2	2	2	2	3	3	3															
Valongo	1	1	2	2	2	2	2	3	3																
Suzão	1	1	2	2	2	2	2	3																	
Cabeda	1	1	1	2	2	2																			
Ermesinde	1	1	1	1	2	2																			
Águas Santas / Palmilheira	1	1	1	1	2																				
Rio Tinto	1	1	1																						
Contumil	1	1																							
Porto (Campanhã)	1																								
Porto (S. Bento)																									

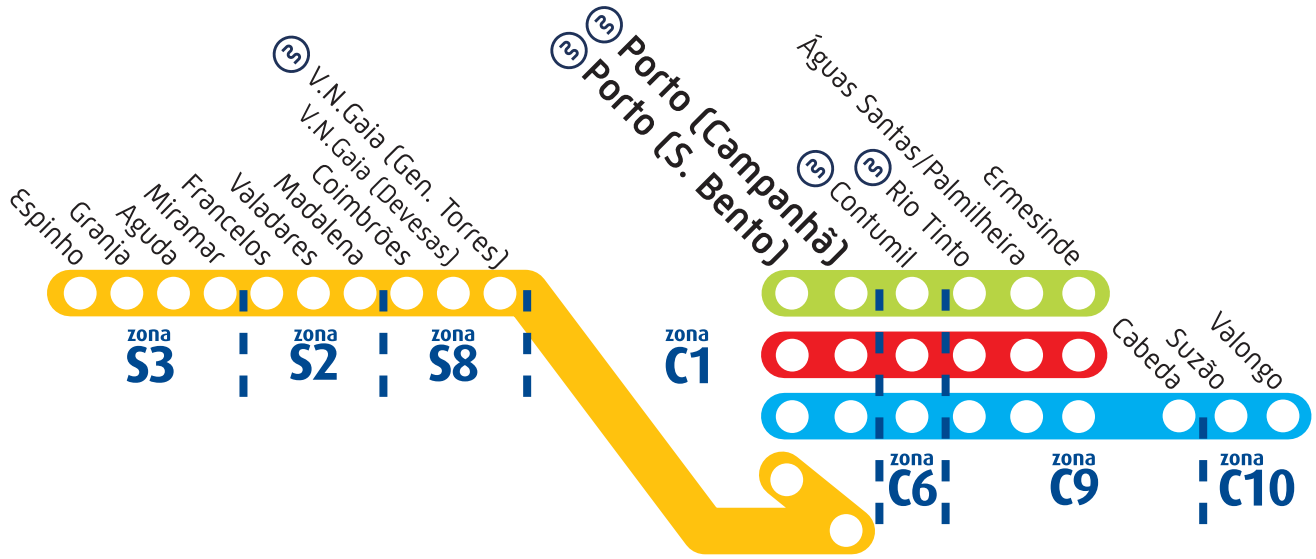
9	9	9	8	8	8	8	8	7	7	7	6	6	6	6	5	5	4	4	3	2	2	2	1	**Aveiro**
8	8	8	8	7	7	7	7	6	6	6	6	5	5	5	5	4	3	3	2	2	1	1		Cacia
8	8	7	7	7	7	6	6	6	6	5	5	5	4	4	4	4	3	2	2	1	1			Canelas
8	7	7	7	7	6	6	6	6	5	5	5	4	4	4	4	3	3	2	2	1				Salreu
7	7	7	6	6	6	6	6	5	5	5	4	4	4	4	3	3	2	2	1					Estarreja
6	6	6	6	5	5	5	5	4	4	4	4	3	3	3	3	2	1	1						Avanca
6	6	5	5	5	5	5	4	4	4	4	3	3	3	2	2	2	1							Válega
6	5	5	5	5	4	4	4	4	3	3	3	2	2	2	2	1								Ovar
5	4	4	4	4	3	3	3	3	2	2	2	1	1	1	1									Carvalh. Maceda
4	4	4	4	3	3	3	3	2	2	2	2	1	1	1										Cortegaça
4	4	3	3	3	3	3	2	2	2	2	1	1	1											Esmoriz
4	4	3	3	3	3	2	2	2	2	1	1	1												Paramos
4	3	3	3	3	2	2	2	2	1	1	1													Silvalde
3	3	3	3	2	2	2	2	1	1	1														Espinho
3	3	2	2	2	2	1	1	1	1															Granja
3	2	2	2	2	2	1	1	1																Aguda
2	2	2	2	1	1	1	1																	Miramar
2	2	2	1	1	1	1																		Francelos
2	2	1	1	1	1																			Valadares
2	1	1	1	1																				Madalena
2	1	1	1																					Coimbrões
1	1	1																						V.N. Gaia (Devesas)
1	1																							V. N. Gaia (Gen. Torres)
1																								Porto (Campanhã)
Porto (S. Bento)																								

COMBOIOS DE PORTUGAL

Comboios Urbanos do Porto

andante

Zonas andante nas linhas da CP Porto



Títulos Ocasionais

	Porto S. Bento	Porto Campanhã	Contumil	Rio Tinto	Águas Santas / Palmilheira	Ermesinde	Cabeda	Suzão	Valongo	G. Torres	V. N. Gaia	Coimbrões	Madalena	Valadares	Francelos	Miramar	Aguda	Granja	Espinho
Porto Campanhã	Z2																		
Contumil	Z2	Z2																	
Rio Tinto	Z2	Z3	Z3																
Águas Santas / Palmilheira	Z2	Z2	Z3	Z3															
Ermesinde	Z2	Z2	Z2	Z3	Z3														
Cabeda	Z2	Z2	Z2	Z2	Z3	Z3													
Suzão	Z2	Z2	Z2	Z2	Z3	Z4	Z4												
Valongo	Z2	Z2	Z2	Z2	Z3	Z4	Z4												
G. Torres	Z4	Z4	Z3	Z3	Z3	Z3	Z2	Z2	Z2										
V. N. Gaia	Z2	Z4	Z4	Z3	Z3	Z3	Z3	Z2	Z2	Z2									
Coimbrões	Z2	Z2	Z4	Z4	Z3	Z3	Z3	Z3	Z2	Z2	Z2								
Madalena	Z2	Z2	Z2	Z5	Z5	Z4	Z4	Z4	Z4	Z3	Z3	Z3							
Valadares	Z2	Z2	Z2	Z2	Z5	Z5	Z4	Z4	Z4	Z4	Z3	Z3	Z3						
Francelos	Z2	Z2	Z2	Z2	Z2	Z5	Z5	Z4	Z4	Z4	Z4	Z3	Z3	Z3					
Miramar	Z2	Z2	Z2	Z3	Z3	Z3	Z6	Z6	Z5	Z5	Z5	Z5	Z4	Z4	Z4				
Aguda	Z2	Z2	Z2	Z2	Z3	Z3	Z3	Z6	Z6	Z5	Z5	Z5	Z5	Z4	Z4	Z4			
Granja	Z2	Z2	Z2	Z2	Z2	Z3	Z3	Z3	Z6	Z6	Z5	Z5	Z5	Z5	Z4	Z4	Z4		
Espinho	Z2	Z2	Z2	Z2	Z2	Z2	Z3	Z3	Z3	Z6	Z6	Z5	Z5	Z5	Z5	Z4	Z4	Z4	

Títulos de Assinatura

	Porto S. Bento	Porto Campanhã	Contumil	Rio Tinto	Águas Santas / Palmilheira	Ermesinde	Cabeda	Suzão	Valongo	G. Torres	V. N. Gaia	Coimbrões	Madalena	Valadares	Francelos	Miramar	Aguda	Granja	Espinho
Porto Campanhã	Z2																		
Contumil	Z2	Z2																	
Rio Tinto	Z3	Z3	Z2																
Águas Santas / Palmilheira	Z3	Z3	Z2	Z2															
Ermesinde	Z3	Z3	Z2	Z2	Z2														
Cabeda	Z3	Z3	Z2	Z2	Z2	Z2													
Suzão	Z4	Z4	Z3	Z2	Z2	Z2	Z2												
Valongo	Z4	Z4	Z3	Z2	Z2	Z2	Z2	Z2											
G. Torres	Z2	Z2	Z3	Z4	Z4	Z4	Z4	Z5	Z5										
V. N. Gaia	Z2	Z2	Z3	Z4	Z4	Z4	Z4	Z5	Z5	Z2									
Coimbrões	Z2	Z2	Z3	Z4	Z4	Z4	Z4	Z5	Z5	Z2	Z2								
Madalena	Z3	Z3	Z4	Z5	Z5	Z5	Z5	Z6	Z6	Z2	Z2	Z2							
Valadares	Z3	Z3	Z4	Z5	Z5	Z5	Z5	Z6	Z6	Z2	Z2	Z2	Z2						
Francelos	Z3	Z3	Z4	Z5	Z5	Z5	Z5	Z6	Z6	Z2	Z2	Z2	Z2	Z2					
Miramar	Z4	Z4	Z5	Z6	Z6	Z6	Z6	Z7	Z7	Z3	Z3	Z3	Z2	Z2	Z2				
Aguda	Z4	Z4	Z5	Z6	Z6	Z6	Z6	Z7	Z7	Z3	Z3	Z3	Z2	Z2	Z2	Z2			
Granja	Z4	Z4	Z5	Z6	Z6	Z6	Z6	Z7	Z7	Z3	Z3	Z3	Z2	Z2	Z2	Z2	Z2		
Espinho	Z4	Z4	Z5	Z6	Z6	Z6	Z6	Z7	Z7	Z3	Z3	Z3	Z2	Z2	Z2	Z2	Z2	Z2	

linhandante
808 200 444
www.linhandante.com
cliente@linhandante.com

References

[1] Mitsuo Gen, Juno Choi, and Kenichi Ida. Improved genetic algorithm for generalized transportation problem. *Artificial Life and Robotics*, 4(2):96–102, 2000.

[2] Público e Lusa. Actividade da stcp pode ficar praticamente confinada à cidade do porto. February 2014. Available at http://www.publico.pt/local/noticia/actividade-da-stcp-pode-ficar-praticamente-confinada-a-cidade-do-porto-1622

[3] Carlos Moura. Barraqueiro defende regime de transição até 2019. February 2014. Available at http://www.transportesemrevista.com/Default.aspx?id=1565&language=pt-PT&tabid=210.

[4] Parlamento Europeu. Regulamento (ce) n.o 1370/2007 do parlamento europeu e do conselho de 23 de outubro de 2007 relativo aos serviços públicos de transporte ferroviário e rodoviário de passageiros e que revoga os regulamentos (cee) n.o 1191/69 e (cee) n.o 1107/70 do conselho. *Jornal Oficial da União Europeia*, December 2000. Available at http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2007:315:0001:0001:PT:PDF.

[5] European Comission. Liberalisation: opening markets to competition - european commission. Availabe at: http://ec.europa.eu/competition/liberalisation/overview_en.html.

[6] Associação Nacional dos Transportadores Rodoviários em Automóveis Ligeiros (ANTRAL) e Federação Portuguesa do Táxi (F.P.T.) Direção-Geral das Atividades Económicas. Convenção para a prestação do serviço de transporte de passageiros em taxi, 2013. Available at: http://antral.pt/pdf/tabela%20de%20precos%202013.PDF.

[7] C. von Ferber, T. Holovatch, Yu. Holovatch, and V. Palchykov. Public transport networks: empirical analysis and modeling. *The European Physical Journal B*, 68(2):261–275, 2009.

[8] Sybil Derrible and Christopher Kennedy. The complexity and robustness of metro networks. *Physica A: Statistical Mechanics and its Applications*, 389(17):3678 – 3691, 2010.

[9] C. von Ferber, Y. Holovatch, and V. Palchykov. Scaling in public transport networks. *eprint arXiv:cond-mat/0501296*, January 2005.

[10] R. Ferrer_i_cancho and R. V. Solé. Optimisation in Complex Networks. In Pastor R. Satorras, M. Rubi, and Díaz A. Guilera, editors, *Statistical Mechanics of Complex Networks Lectures Notes in Physics 625*, volume 625, pages 114–126, 2001.

[11] Sara Maia Ribeiro. Sistemas de tarifários em transportes públicos de passageiros. Master's thesis, Faculdade de Engenharia da Universidade do Porto, 2011.

[12] SPUTNIC (Strategies for Public Transport in Cities). Guidelines in market organisation - public transport integration. Available at: http://www.sputnicproject.eu/docs/Sputnic-ptintegration.pdf.

[13] Joshua J. Maciejewski. Automating journey fare calculation for transport for london. Master's thesis, Massachusetts Institute of Technology, June 2008.

[14] Transport for London. Adult rate tube, dlr and london overground fares, January 2014. Available at: http://www.tfl.gov.uk/assets/downloads/tube-dlr-lo-adult-fares-jan-2014.pdf.

[15] O andante – curso rápido de introdução, April 2013.

[16] Ravindra K. Ahuja, Kurt Mehlhorn, James Orlin, and Robert E. Tarjan. Faster algorithms for the shortest path problem. *J. ACM*, 37(2):213–223, April 1990.

[17] Stefano Pallottino and Maria Grazia Scutellà. Shortest path algorithms in transportation models: Classical and innovative aspects, 1998.

[18] P. Biswas, P. K. Mishra, and N. C. Mahanti. Computational efficiency of optimized shortest path algorithms. *IJCSA*, 2(2):22–37, 2005.

[19] F. Benjamin Zhan and Charles E. Noon. Shortest path algorithms: An evaluation using real road networks. *Transportation Science*, 32(1):65–73, 1998.

[20] AndrewV. Goldberg. Optimization algorithms for large networks. In Jan Leeuwen, editor, *Algorithms — ESA '94*, volume 855 of *Lecture Notes in Computer Science*, pages 1–9. Springer Berlin Heidelberg, 1994.

[21] Timothy M. Chan. More algorithms for all-pairs shortest paths in weighted graphs. In *In Proceedings of 39th Annual ACM Symposium on Theory of Computing*, pages 590–598, 2007.

[22] Yijie Han. An $o(n^3(log(log(n)/log^2(n))))$ time algorithm for all pairs shortest paths. In *Proceedings of the 14th Conference on Annual European Symposium - Volume 14*, ESA'06, pages 411–417, London, UK, UK, 2006. Springer-Verlag.

[23] Wei Peng, Xiaofeng Hu, Feng Zhao, and Jinshu Su. A fast algorithm to find all-pairs shortest paths in complex networks. *Procedia Computer Science*, 9(0):557 – 566, 2012. Proceedings of the International Conference on Computational Science, {ICCS} 2012.

[24] D. T. Lee. Interval, segment, range and priority search trees. In Dinesh Mehta and Sartaj Sahni, editors, *The Handbook of Data Structures and Applications*, chapter 34, pages 18–1 18–21. 2005.

[25] Douglas Comer. Ubiquitous b-tree. *ACM Comput. Surv.*, 11(2):121–137, June 1979.

[26] Christin Wiedemann. Applying the scientific method to software testing. *SearchSoftwareQuality*, October 2013. Available at http://searchsoftwarequality.techtarget.com/feature/Applying-the-scientific-method-to-software-testing.

[27] S. N. Dorogovtsev and J. F. F. Mendes. Evolution of networks. *Advances in Physics*, 51(4):1079–1187, 2002.

[28] vis.js. vis.js | a dynamic, browser-based visualization library. Available at http://visjs.org/.

[29] Jerald Jariyasunant. Algorithm for finding optimal paths in a public transit network with real-time data. November 2010.

[30] Marco BRAMBILLA Angelo MARTINO MKmetric Dr. Benedikt MANDEL Oliver SCHNELL TRT Trasporti e Territorio Silvia MAFFII, Alessio SITRAN. Integrated ticketing on long-distance passanger transport services. August 2013. Available at http://www.europarl.europa.eu/RegData/etudes/etudes/join/2012/474566/IPOL-TRAN_ET(2012)474566_EN.pdf.